Z80 MICROCOMPUTER SYSTEM

Z80

SGS

Z80™ MICROCOMPUTER SYSTEM

EUROCRATIC MOS

Since setting up its MOS department in 1966, SGS-ATES has led the way in European MOS technology.

Between the major landmarks of the first European-designed MOS calculator in 1968 and the F8 microprocessor in 1977, we brought you a full range of memories: 1K static and 4K dynamic RAMs, a 1K x 8 EPROM, a 2K x 8 ROM.....
and now we bring you the Z-80.

Not only the Z-80 but a team of experts dedicated to the development of the Z-80 device family, Z-80 systems, applications and interface devices.

Moreover, we've set up a comprehensive European network of "local" microcomputer application centres packed with the most up-to-date equipment available, staffed with highly-experienced software engineers and located in UK, Sweden, Italy, France and Germany.

SGS-ATES and Zilog: a vast reserve of know-how and resources committed to the advancement of microprocessors - stay with us and be part of the Z-80 conquest.

CONTENTS

Z-80 MICROCOMPUTER PRODUCT LINE

Introduction

The Z-80 LSI component set includes all of the logic circuits necessary for the user to build high performance microcomputer systems with virtually no external logic and an absolute minimum number of lowest cost standard memory components. The Z-80 component set is backed by advanced software, a disk based hardware/software development system and complete training and support. The entire Z-80 product line has been developed as a single, highly integrated entity to insure that the user can develop his system quickly and still obtain all the performance advantages of the Z-80 component set.

High System Throughput

The architecture of the Z-80 CPU includes a superset of 158 instructions, with more internal registers and addressing modes than second generation microcomputers and extremely fast interrupt response time. All of these features mean that in any given amount of time the Z-80 can perform *far* more work (processor throughput) than any other micro-computer system available today. This throughput advantage allows users to continually expand the features and capabilities of their systems without increasing hardware costs.

Low Memory Costs

One of the major features of the Z-80 CPU is that it greatly reduces system memory costs. The expanded set of 158 software instructions results in a tremendous reduction in the memory required for any typical application. In addition, the Z-80 CPU provides all refresh and timing signals to directly drive dynamic memories so that the Z-80 LSI components can interface to most standard 4K dynamic memories with virtually no external logic. The Z-80 CPU uses a technique whereby the memory address is generated very early in memory cycles, permitting the high speed Z-80 CPU to operate with standard speed memories, again reducing system memory costs. The Z-80 CPU was designed to operate with standard memory products from any source since these devices will always be less expensive than custom memories designed for any particular microcomputer.

Low I/O Costs

The Z-80 LSI component set includes four general purpose programmable I/O circuits that contain all of the logic required to implement fast I/O transfers with minimal CPU overhead. These circuits have a built-in ripple priority interrupt control circuit (the device closest to the CPU has the highest priority) and all the logic necessary for nesting of interrupts to any level. Using the programmable features of these circuits, the user can configure the devices to interface with a wide range of peripheral devices with virtually no other external logic. These features make the peripheral device controllers in a Z-80 system much simpler and therefore lower in cost.

Low System Hardware Costs

The Z-80 component set requires very little support circuitry. All devices require a single +5 volt power supply and a single phase TTL clock. In addition, all control signals are directly compatible with I/O and memory devices so that system control circuits are not required. External interrupt control circuits are not required since these are included in every Z-80 I/O circuit. DMA circuits are generally not required due to an extremely fast interrupt response and powerful I/O block transfer capability within the CPU.

Low Development Costs

SGS-ATES offers more than a fully integrated line of LSI components. Everything is provided that is necessary for the user to easily develop his own proprietary system using the Z-80 components. This includes complete software packages, disk based development systems and training. For example, the expanded Z-80 software instruction set coupled with the easy to learn Z-80 assembly language and reference cards make assembly language programming much easier than previously possible. For larger programs, PL/Z may be used to speed up the development cycle, to enhance program documentation and to improve program maintainability.

Z-80 MICROCOMPUTER SUMMARY

Central Processor Unit/Z-80-CPU

- ☐ Single chip, N-channel processor
- ☐ 158 instructions - Includes all 78 of the 8080A instructions with *total* software compatibility. New instructions include 4-, 8- and 16-bit operations with more useful addressing modes.
- ☐ 17 internal registers (more than twice the 8080A registers), including two real index registers.
- ☐ Three modes of fast interrupt response plus a nonmaskable interrupt.
- ☐ Directly interfaces standard speed static or dynamic memories with virtually no external logic.
- ☐ 1.6μs instruction execution speed.
- ☐ Single 5V supply and single-phase TTL Clock.
- ☐ Out-performs any other microcomputer in 4-4-, 8-, 16-bit applications.
- ☐ Requires 25% to 50% less memory space than the 8080A CPU.
- ☐ Up to 500% more throughput than the 8080A.
- ☐ TTL compatible tri-state data and address busses.

Interface and Control Circuits

Parallel Input/Output Controller/Z-80-PIO

Programmable circuit that allows for a direct interface to a wide range of parallel interface peripherals without other external logic.

Serial Input/Output Controller/Z-80-SIO

Programmable circuit that allows for a direct interface to a wide range of serial interface peripherals without other external logic.

Counter Timer Circuit/Z-80-CTC

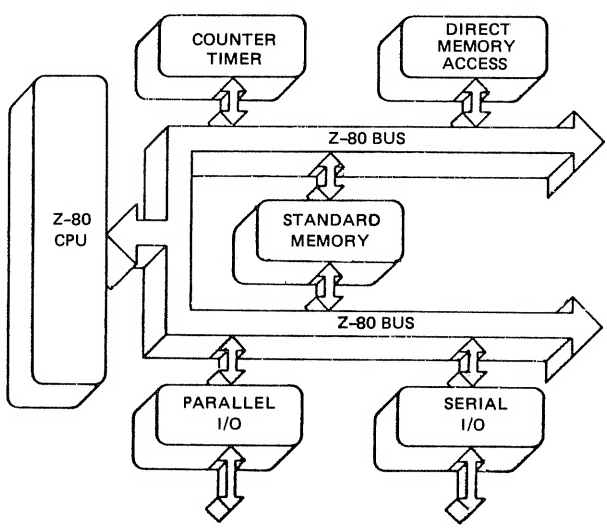
Contains four independent programmable counter timer circuits for control of real time events.

Direct Memory Access Controller/Z-80-DMA

Programmable circuit that can directly transfer data between the SIO or PIO and memory on a CPU cycle steal basis.

All Z-80 controllers have built in nested priority interrupt control and fast interrupt response capability (up to 6 times faster than the 8080A).

All Z-80 controllers monitor peripheral status to eliminate any type of CPU polling.



Z-80 COMPONENTS

Introduction

The SGS-ATES third generation microcomputer components are the most advanced and comprehensive set of LSI microcomputer products available today. The major components in the Z-80 product line are an extremely high performance central processing unit (CPU), a programmable parallel input/output controller (PIO), a programmable serial input/output controller (SIO), a versatile counter timer circuit (CTC) and a high speed direct memory access controller (DMA).

All of the Z-80 components utilize the industry standard N-channel silicon gate technology to provide the highest density at the lowest cost. Depletion load technology is also used to provide high performance with a single 5V power supply.

The CPU, PIO, SIO and DMA are packages in standard 40-pin DIPs; the CTC comes in a standard 28-pin DIP. All require only a single 5V power supply plus the Z-80 single-phase TTL level clock.

Z-80 CPU

The Z-80 CPU is an extremely powerful, third generation CPU which incorporates a number of major features over the standard 8080A CPU while retaining total software compatibility. Major improvements include: ☐ More than twice as many registers on the CPU chip, including two real index registers ☐ Many more addressing modes ☐ More than twice as many instructions ☐ Three modes of extremely fast interrupt response ☐ A separate non-maskable interrupt to a fixed location.

Another unique feature of the Z-80 CPU is its ability to generate all of the control signals for standard memory circuits. Static memories can be interfaced using only an external address decoder for chip selects. In addition the Z-80 CPU provides all of the refresh control for dynamic memories, and the Z-80 control bus timing signals are directly compatible with all widely used, standard speed, 18- and 22-pin 4K RAMs (16-pin 4K RAMs require only an external address multiplexer). Thus dynamic RAMs can be interfaced with virtually no additional external logic. This provides the user with the ability to easily interface to the lowest cost dynamic memories without reducing CPU operational speed.

By selecting the best standard memory for a given application, the user can reduce his product manufacturing costs, and the product development expenses will also be much lower.

The Z-80 CPU is designed to be totally software compatible with the standard 8080A microprocessor to facilitate the user's transition to the Z-80. By using the Z-80 component set and the most economical memory for the particular application, the user need only relayout any 8080 based design and use any existing software programs to obtain an immediate and very significant reduction in system hardware costs. A major advantage is that the same ROMs that are used in the 8080 system can be used in the Z-80 system. At a later date the software programs can be upgraded, taking advantage of the powerful Z-80 instruction set and the full capability of the Z-80 component set to obtain increased performance and even further cost reduction for memory components.

The Z-80 CPU is an extremely fast and versatile device. Full instruction cycle times for non-memory reference instructions are 1.6μs and the CPU responds to interrupts very rapidly (the 8080 requires up to 6 times as long to respond, and uses more than twice as much memory storage). This fast interrupt response, in conjunction with new I/O block transfer instructions, allows the CPU to directly control many peripherals without the costly use of DMA hardware and it greatly reduces the size of software routines required for peripheral control, again saving memory space and costs.

Probably the most important feature of the Z-80 microprocessor family is its repertoire of 158 software instructions. The original 78 instructions of the 8080A CPU are included using the same OP codes; thus, the Z-80 can execute 8080 or 8080A programs stored in existing ROMs. The Z-80 new software instructions provide an expanded capability for the user, such as: ☐ Additional addressing modes, including indexed and relative ☐ Memory to memory block transfers and searches ☐ Bit manipulation and testing in any register or memory location ☐ Many new I/O instructions, including block I/O transfers ☐ A wide range of memory or register rotates and shifts (logical and arithmetic) ☐ Expanded 16-bit arithmetic ☐ Expanded BCD arithmetic.

Parallel Input/Output (PIO)

The Z-80 PIO circuit uses an advanced interrupt driven, program controlled I/O transfer technique for easy handling of virtually any peripheral with a parallel interface. Without other logic, the PIO can interface most line printers, paper tape readers or punchers, card readers, keyboards, electronic typewriters and other similar devices.

The PIO contains all of the interrupt control logic necessary for nested priority interrupt handling with very fast response time. Thus additional interrupt control circuits are not needed and servicing time is minimized. The parallel I/O can handle two high speed I/O ports, and it interrupts the CPU after each I/O transfer is complete.

The PIO circuit include two independent ports, each with eight I/O lines and two handshake lines which are programmed by the CPU to operate in one of four modes: □ Byte output with interrupt driven handshake □ Byte input with interrupt driven handshake □ Bidirectional byte bus with interrupt driven handshake □ Control mode wherein any bit can be programmed as an input or output.

A major feature of the PIO is its ability to generate an interrupt on any bit pattern at the I/O pins, thus eliminating the need for the processor to constantly test I/O lines for a particular peripheral status condition. This feature greatly enhances the ability of the processor to easily handle peripherals, while also reducing software overhead.

Serial Input/Output (SIO)

The SIO circuit is a programmable I/O device similar in concept to the PIO, except that it is designed to handle peripherals with a serial data interface such as floppy disks, CRTs and communication terminals. Each SIO circuit can handle a full duplex serial I/O channel. The device will handle data that is asynchronous with 5- to 8-bit characters and with 1, 1 $\frac{1}{2}$ or 2 stop bits. The SIO will handle 5- to 8-bit synchronous data including IBM BiSync and SDL communication channels. CRC generation and parity checking are also included.

Counter Timer Circuit (CTC)

The CTC circuit contains four versatile clocks, each with its own nested priority interrupt control. All clocks have a minimum resolution of 8μs and can generate interrupts in the range of 8μs to 32 ms. The circuit may also be used in a mode in which it counts external events. Another major feature is that an interrupt can be programmed to occur after the occurrence of an external event. The four timing circuits greatly ease the CPU software handling requirements for many real-time control applications. For example, the CTC allows the implementation of a very low-cost TTY or CRT I/O port, and simple sector control of floppy disk subsystems.

Direct Memory Access Controller (DMA)

The DMA circuit is provided for those applications in which data must be transferred directly into memory at a very high rate rather than going through the central processor unit. This circuit is not needed for most applications due to the fast interrupt response and block transfer capabilities of the Z-80 CPU. However, in large systems applications with many high speed peripherals, such as floppy disks, communications channels, etc., the DMA circuit can greatly improve system performance by totally controlling block transfers between I/O circuits and the system memory.

The DMA circuit contains all control for four I/O circuits including a block length counter and a memory address pointer. The circuits also have a ripple priority chain so that virtually any number of DMA channels can be implemented. The DMA circuit communicates directly between the I/O circuits and the system memory after obtaining a DMA acknowledge signal from the CPU.

Z 80-CPU
Z 80A-CPU

Product Specification

The SGS-ATES Z80 product line is a complete set of microcomputer components, development systems and support software. The Z80 microcomputer component set includes all of the circuits necessary to build high-performance microcomputer systems with virtually no other logic and a minimum number of low cost standard memory elements.

The Z80 and Z80A CPU's are third generation single chip microprocessors with unrivaled computational power. This increased computational power results in higher system through-put and more efficient memory utilization when compared to second generation microprocessors. In addition, the Z80 and Z80A CPU's are very easy to implement into a system because of their single voltage requirement plus all output signals are fully decoded and timed to control standard memory or peripheral circuits. The circuit is implemented using an N-channel, ion implanted, silicon gate MOS process.

Figure 1 is a block diagram of the CPU, Figure 2 details the internal register configuration which contains 208 bits of Read/Write memory that are accessible to the programmer. The registers include two sets of six general purpose registers that may be used individually as 8-bit registers or as 16-bit register pairs. There are also two sets of accumulator and flag registers. The programmer has access to either set of main or alternate registers through a group of exchange instructions. This alternate set allows foreground/background mode of operation or may be reserved for very fast Interrupt response. Each CPU also contains a 16-bit stack pointer which permits simple implementation of multiple level interrupts, unlimited subroutine nesting and simplification of many types of data handling.

The two 16-bit index registers allow tabular data manipulation and easy implementation of relocatable code. The Refresh register provides for automatic, totally transparent refresh of external dynamic memories. The I register is used in a powerful interrupt response mode to form the upper 8 bits of a pointer to a interrupt service address table, while the interrupting device supplies the lower 8 bits of the pointer. An indirect call is then made to this service address.

FEATURES

- Single chip, N-channel Silicon Gate CPU.
- 158 instructions–includes all 78 of the 8080A instructions with total software compatibility. New instructions include 4-, 8- and 16-bit operations with more useful addressing modes such as indexed, bit and relative.
- 17 internal registers.
- Three modes of fast interrupt response plus a non-maskable interrupt.
- Directly interfaces standard speed static or dynamic memories with virtually no external logic.
- 1.0 μs instruction execution speed.
- Single 5 VDC supply and single-phase 5 volt Clock.
- Out-performs any other single chip microcomputer in 4-, 8-, or 16-bit applications.
- All pins TTL Compatible
- Built-in dynamic RAM refresh circuitry.

Fig. 1 - Z80, Z80A CPU BLOCK DIAGRAM

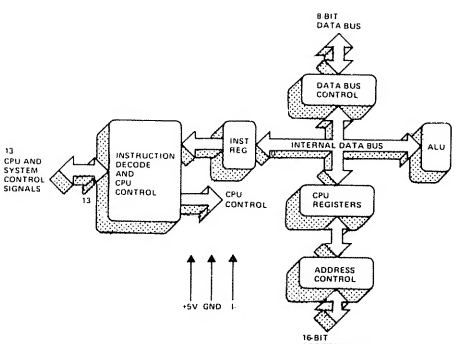
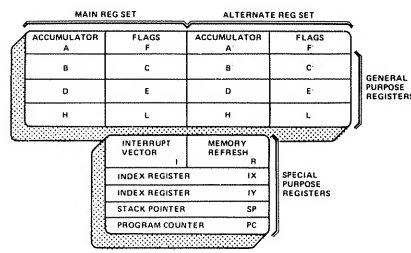
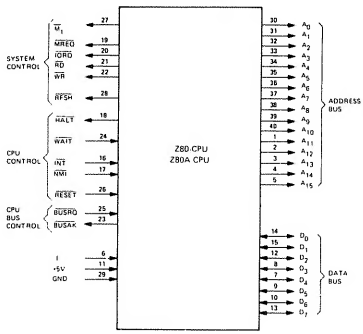


Fig. 2 - Z80, Z80A CPU REGISTERS



Pin Description

PIN CONFIGURATION



A_0-A_{15} (Address Bus) — Tri-state output, active high. A_0-A_{15} constitute a 16-bit address bus. The address bus provides the address for memory (up to 64K bytes) data exchanges and for I/O device data exchanges.

D_0-D_7 (Data Bus) — Tri-state input/output, active high. D_0-D_7 constitute an 8-bit bidirectional data bus. The data bus is used for data exchanges with memory and I/O devices.

$\overline{M_1}$ (Machine Cycle one) — Output, active low. $\overline{M_1}$ indicates that the current machine cycle is the OP code fetch cycle of an instruction execution.

$\overline{MREQ}$ (Memory Request) — Tri-state output, active low. The memory request signal indicates that the address bus holds a valid address for a memory read or memory write operation.

$\overline{IORQ}$ (Input/ Output Request) — Tri-state output, active low. The $\overline{IORQ}$ signal indicates that the lower half of the address bus holds a valid I/O address for a I/O read or write operation. An $\overline{IORQ}$ signal is also generated when an interrupt is being acknowledged to indicate that an interrupt response vector can be placed on the data bus.

$\overline{RD}$ (Memory Read) — Tri-state output, active low. $\overline{RD}$ indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

$\overline{WR}$ (Memory Write) — Tri-state output, active low. $\overline{WR}$ indicates that the CPU data bus holds valid data to be stored in the addressed memory or I/O device.

$\overline{RFSH}$ (Refresh) — Output, active low. $\overline{RFSH}$ indicates that the lower 7 bits of the address bus contain a refresh address for dynamic memories and the current $\overline{MREQ}$ signal should be used to do a refresh read to all dynamic memories.

$\overline{HALT}$ (Halt state) — Output, active low. $\overline{HALT}$ indicates that the CPU has executed a HALT software instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can resume. While halted, the CPU executes NOP's to maintain memory refresh activity.

$\overline{WAIT}$ (Wait) — Input, active low. $\overline{WAIT}$ indicates to the Z-80 CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter wait states for as long as this signal is active.

$\overline{INT}$ (Interrupt Request) — Input, active low. The Interrupt Request signal is generated by I/O devices. A request will be honored at the end of the current instruction if the internal software controlled interrupt enable flip-flop (IFF) is enabled.

$\overline{NMI}$ (Non Maskable Interrupt) — Input, active low. The non-maskable interrupt request line has a higher priority than $\overline{INT}$ and is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop. $\overline{NMI}$ automatically forces the Z-80 CPU to restart to location 0066_H.

$\overline{RESET}$ — Input, active low. $\overline{RESET}$ initializes the CPU as follows: reset interrupt enable flip-flop, clear PC and registers I and R and set interrupt to 8080A mode. During reset time, the address and data bus go to a high impedance state and all control output signals go to the inactive state.

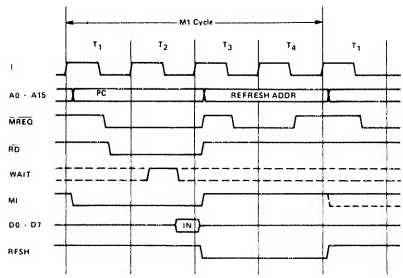
$\overline{BUSRQ}$ (Bus Request) — Input, active low. The bus request signal has a higher priority than $\overline{NMI}$ and is always recognized at the end of the current machine cycle and is used to request the CPU address bus, data bus and tri-state output control signals to go to a high impedance state so that other devices can control these busses.

$\overline{BUSAK}$ (Bus Acknowledge) — Output, active low. Bus acknowledge is used to indicate to the requesting device that the CPU address bus, data bus and tri-state control bus signals have been set to their high impedance state and the external device can now control these signals.

Timing Waveforms

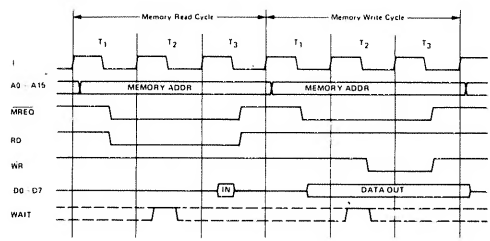
INSTRUCTION OP CODE FETCH

The program counter content (PC) is placed on the address bus immediately at the start of the cycle. One half clock time later $\overline{MREQ}$ goes active. The falling edge of $\overline{MREQ}$ can be used directly as a chip enable to dynamic memories. $\overline{RD}$ when active indicates that the memory data should be enabled onto the CPU data bus. The CPU samples data with the rising edge of the clock state T_3. Clock states T_3 and T_4 of a fetch cycle are used to refresh dynamic memories while the CPU is internally decoding and executing the instruction. The refresh control signal $\overline{RFSH}$ indicates that a refresh read of all dynamic memories should be accomplished.



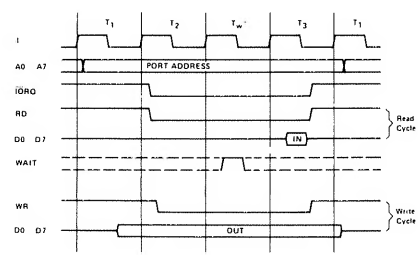
MEMORY READ OR WRITE CYCLES

Illustrated here is the timing of memory read or write cycles other than an OP code fetch (M_1 cycle). The $\overline{MREQ}$ and $\overline{RD}$ signals are used exactly as in the fetch cycle. In the case of a memory write cycle, the $\overline{MREQ}$ also becomes active when the address bus is stable so that it can be used directly as a chip enable for dynamic memories. The $\overline{WR}$ line is active when data on the data bus is stable so that it can be used directly as a R/W pulse to virtually any type of semiconductor memory.



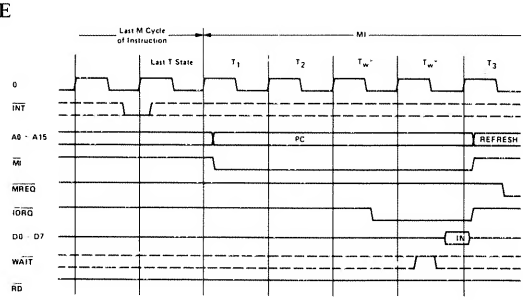
INPUT OR OUTPUT CYCLES

Illustrated here is the timing for an I/O read or I/O write operation. Notice that during I/O operations a single wait state is automatically inserted (Tw*). The reason for this is that during I/O operations this extra state allows sufficient time for an I/O port to decode its address and activate the $\overline{WAIT}$ line if a wait is required.



INTERRUPT REQUEST/ACKNOWLEDGE CYCLE

The interrupt signal is sampled by the CPU with the rising edge of the last clock at the end of any instruction. When an interrupt is accepted, a special M_1 cycle is generated. During this M_1 cycle, the $\overline{IORQ}$ signal becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. Two wait states (Tw*) are automatically added to this cycle so that a ripple priority interrupt scheme, such as the one used in the Z80 peripheral controllers, can be easily implemented.



Instruction Set

The following is a summary of the Z80, Z80A instruction set showing the assembly language mnemonic and the symbolic operation performed by the instruction. A more detailed listing appears in the Z80-CPU technical manual, and assembly language programming manual. The instructions are divided into the following categories:

8-bit loads	Miscellaneous Group
16-bit loads	Rotates and Shifts
Exchanges	Bit Set, Reset and Test
Memory Block Moves	Input and Output
Memory Block Searches	Jumps
8-bit arithmetic and logic	Calls
16-bit arithmetic	Restarts
General purpose Accumulator & Flag Operations	Returns

In the table the following terminology is used.

b ≡ a bit number in any 8-bit register or memory location

cc ≡ flag condition code
- NZ ≡ non zero
- Z ≡ zero
- NC ≡ non carry
- C ≡ carry
- PO ≡ Parity odd or no over flow
- PE ≡ Parity even or over flow
- P ≡ Positive
- M ≡ Negative (minus)

d ≡ any 8-bit destination register or memory location

dd ≡ any 16-bit destination register or memory location

e ≡ 8-bit signed 2's complement displacement used in relative jumps and indexed addressing

L ≡ 8 special call locations in page zero. In decimal notation these are 0, 8, 16, 24, 32, 40, 48 and 56

n ≡ any 8-bit binary number

nn ≡ any 16-bit binary number

r ≡ any 8-bit general purpose register (A, B, C, D, E, H, or L)

s ≡ any 8-bit source register or memory location

s_b ≡ a bit in a specific 8-bit register or memory location

ss ≡ any 16-bit source register or memory location

subscript "L" ≡ the low order 8 bits of a 16-bit register

subscript "H" ≡ the high order 8 bits of a 16-bit register

( ) ≡ the contents within the ( ) are to be used as a pointer to a memory location or I/O port number

8-bit registers are A, B, C, D, E, H, L, I and R

16-bit register pairs are AF, BC, DE and HL

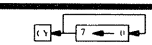
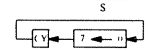
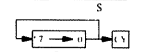
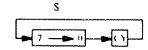
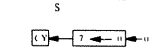
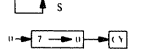
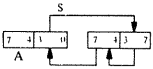
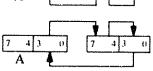
16-bit registers are SP, PC, IX and IY

Addressing Modes implemented include combinations of the following.

Immediate	Indexed
Immediate extended	Register
Modified Page Zero	Implied
Relative	Register Indirect
Extended	Bit

	Mnemonic	Symbolic Operation	Comments
8-BIT LOADS	LD r, s	r ← s	s ≡ r, n, (HL), (IX+e), (IY+e)
	LD d, r	d ← r	d ≡ (HL), r (IX+e), (IY+e)
	LD d, n	d ← n	d ≡ (HL), (IX+e), (IY+e)
	LD A, s	A ← s	s ≡ (BC), (DE), (nn), I, R
	LD d, A	d ← A	d ≡ (BC), (DE), (nn), I, R
16-BIT LOADS	LD dd, nn	dd ← nn	dd ≡ BC, DE, HL, SP, IX, IY
	LD dd, (nn)	dd ← (nn)	dd ≡ BC, DE, HL, SP, IX, IY
	LD (nn), ss	(nn) ← ss	ss ≡ BC, DE, HL, SP, IX, IY
	LD SP, ss	SP ← ss	ss = HL, IX, IY
	PUSH ss	$(SP-1) \leftarrow ss_H$; $(SP-2) \leftarrow ss_L$	ss = BC, DE, HL, AF, IX, IY
	POP dd	$dd_L \leftarrow (SP)$; $dd_H \leftarrow (SP+1)$	dd = BC, DE, HL, AF, IX, IY
EXCHANGES	EX DE, HL	DE ↔ HL	
	EX AF, AF'	AF ↔ AF'	
	EXX	$\begin{pmatrix} BC \\ DE \\ HL \end{pmatrix} \leftrightarrow \begin{pmatrix} BC' \\ DE' \\ HL' \end{pmatrix}$	
	EX (SP), ss	$(SP) \cdot\cdot ss_L$, $(SP+1) \cdot\cdot ss_H$	ss ≡ HL, IX, IY

	Mnemonic	Symbolic Operation	Comments
MEMORY BLOCK MOVES	LDI	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1	
	LDIR	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1 Repeat until BC = 0	
	LDD	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1	
	LDDR	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1 Repeat until BC = 0	
MEMORY BLOCK SEARCHES	CPI	A-(HL), HL ← HL+1 BC ← BC-1	
	CPIR	A-(HL), HL ← HL+1 BC ← BC-1, Repeat until BC = 0 or A = (HL)	A-(HL) sets the flags only A is not affected
	CPD	A-(HL), HL ← HL-1 BC ← BC-1	
	CPDR	A-(HL), HL ← HL-1 BC ← BC-1, Repeat until BC= 0 or A = (HL)	
8-BIT ALU	ADD A, s	A ← A + s	
	ADC A, s	A ← A + s + CY	CY is the carry flag
	SUB s	A ← A - s	
	SBC A, s	A ← A - s - CY	s ≡ r, n, (HL) (IX+e), (IY+e)
	AND s	A ← A ∧ s	
	OR s	A ← A ∨ s	
	XOR s	A ← A ⊕ s	

	Mnemonic	Symbolic Operation	Comments
8-Bit ALU	CP s	A - s	s = r, n (HL) (IX+e), (IY+e)
	INC d	d ← d + 1	d = r, (HL) (IX+e), (IY+e)
	DEC d	d ← d - 1	
16-BIT ARITHMETIC	ADD HL, ss	HL ← HL + ss	ss ≡ BC, DE HL, SP
	ADC HL, ss	HL ← HL + ss + CY	
	SBC HL, ss	HL ← HL - ss - CY	
	ADD IX, ss	IX ← IX + ss	ss ≡ BC, DE, IX, SP
	ADD IY, ss	IY ← IY + ss	ss ≡ BC, DE, IY, SP
	INC dd	dd ← dd + 1	dd ≡ BC DE HL, SP, IX, IY
	DEC dd	dd ← dd - 1	dd ≡ BC, DE, HL, SP, IX, IY
GP ACC. & FLAG	DAA	Converts A contents into packed BCD following add or subtract	Operands must be in packed BCD format
	CPL	$A \leftarrow \overline{A}$	
	NEG	A ← 00 - A	
	CCF	$CY \leftarrow \overline{CY}$	
	SCF	CY ← 1	
MISCELLANEOUS	NOP	No operation	
	HALT	Halt CPU	
	DI	Disable Interrupts	
	EI	Enable Interrupts	
	IM 0	Set interrupt mode 0	8080A mode
	IM 1	Set interrupt mode 1	Call to 0038_H
	IM 2	Set interrupt mode 2	Indirect Call
ROTATES AND SHIFTS	RLC s		s ≡ r (HL) (IX+e), (IY+e)
	RL s		
	RRC s		
	RR s		
	SLA s		
	SRA s		
	SRL s		
	RLD		
	RRD		

	Mnemonic	Symbolic Operation	Comments
BIT S, R, & T	BIT b, s	$Z \leftarrow \overline{s_b}$	Z is zero flag
	SET b, s	$s_b \leftarrow 1$	s ≡ r, (HL) (IX+e), (IY+e)
	RES b, s	$s_b \leftarrow 0$	
INPUT AND OUTPUT	IN A, (n)	A ← (n)	
	IN r, (C)	r ← (C)	Set flags
	INI	(HL) ← (C), HL ← HL + 1 B ← B - 1	
	INIR	(HL) ← (C), HL ← HL + 1 B ← B - 1 Repeat until B = 0	
	IND	(HL) ← (C), HL ← HL - 1 B ← B - 1	
	INDR	(HL) ← (C), HL ← HL - 1 B ← B - 1 Repeat until B = 0	
	OUT(n), A	(n) ← A	
	OUT(C), r	(C) ← r	
	OUTI	(C) ← (HL), HL ← HL + 1 B ← B - 1	
	OTIR	(C) ← (HL), HL ← HL + 1 B ← B - 1 Repeat until B = 0	
	OUTD	(C) ← (HL), HL ← HL - 1 B ← B - 1	
	OTDR	(C) ← (HL), HL ← HL - 1 B ← B - 1 Repeat until B = 0	
JUMPS	JP nn	PC ← nn	cc ≡ NZ PO, Z PE, NC P, C M
	JP cc, nn	If condition cc is true PC ← nn, else continue	
	JR e	PC ← PC + e	kk ≡ NZ NC, Z C
	JR kk, e	If condition kk is true PC ← PC + e, else continue	
	JP (ss)	PC ← ss	ss = HL, IX, IY
	DJNZ e	B ← B - 1, if B = 0 continue, else PC ← PC + e	
CALLS	CALL nn	$(SP-1) \leftarrow PC_H$, $(SP-2) \leftarrow PC_L$, PC ← nn	cc ≡ NZ PO, Z PE, NC P, C M
	CALL cc, nn	If condition cc is false continue, else same as CALL nn	
RESTARTS	RST L	$(SP-1) \leftarrow PC_H$, $(SP-2) \leftarrow PC_L$, $PC_H \leftarrow 0$, $PC_L \leftarrow L$	
RETURNS	RET	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$	cc ≡ NZ PO, Z PE, NC P, C M
	RET cc	If condition cc is false continue, else same as RET	
	RETI	Return from interrupt, same as RET	
	RETN	Return from non-maskable interrupt	

Z80-CPU A.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= +5V ± 5%, Unless Otherwise Noted.

Signal	Symbol	Parameter	Min.	Max.	Unit	Test Condition
Φ	t_c	Clock Period	4	[12]	μsec	
	t_{w} (Φ H)	Clock Pulse Width, Clock High	180	[E]	nsec	
	t_{w} (Φ L)	Clock Pulse Width, Clock Low	180	2000	nsec	
	t_{r}, t_f	Clock Rise and Fall Time		30	nsec	
A_{0-15}	t_{D} (AD)	Address Output Delay		145	nsec	
	t_F (AD)	Delay to Float		110	nsec	
	t_{acm}	Address Stable Prior to $\overline{MREQ}$ (Memory Cycle)	[1]		nsec	C_L= 50 pF
	t_{act}	Address Stable Prior to $\overline{IORQ}$, $\overline{RD}$ or $\overline{WR}$ (I/O Cycle)	[2]		nsec	
	t_{ca}	Address Stable from $\overline{RD}$, $\overline{WR}$, $\overline{IORQ}$ or $\overline{MREQ}$	[3]		nsec	
	t_{caf}	Address Stable From $\overline{RD}$ or $\overline{WR}$ During Float	[4]		nsec	
D_{0-7}	t_{D} (D)	Data Output Delay		230	nsec	
	t_F (D)	Delay to Float During Write Cycle		90	nsec	
	$t_{S\Phi}$ (D)	Data Setup Time to Rising Edge of Clock During M1 Cycle	50		nsec	
	$t_{S\Phi}$ (D)	Data Setup Time to Falling Edge of Clock During M2 to M5	60		nsec	C_L= 50 pF
	t_{dcm}	Data Stable Prior to $\overline{WR}$ (Memory Cycle	[5]		nsec	
	t_{dci}	Data Stable Prior to $\overline{WR}$ (I/O Cycle)	[6]		nsec	
	t_{cdf}	Data Stable From $\overline{WR}$	[7]			
	t_H	Any Hold Time for Setup Time	0		nsec	
$\overline{MREQ}$	$t_{DL\overline{\Phi}}$ (MR)	$\overline{MREQ}$ Delay From Falling Edge of Clock $\overline{MREQ}$ Low		100	nsec	
	$t_{DH\Phi}$ (MR)	$\overline{MREQ}$ Delay From Rising Edge of Clock $\overline{MREQ}$ High		100	nsec	
	$t_{DH\overline{\Phi}}$ (MR)	$\overline{MREQ}$ Delay From Falling Edge of Clock $\overline{MREQ}$ High		100	nsec	C_L = 50pF
	t_{w} ($\overline{MRL}$)	Pulse Width, $\overline{MREQ}$ Low	[8]		nsec	
	t_{w} ($\overline{MRH}$)	Pulse Width, $\overline{MREQ}$ High	[9]		nsec	
$\overline{IORQ}$	$t_{DL\Phi}$ (IR)	$\overline{IORQ}$ Delay From Rising Edge of Clock $\overline{IORQ}$ Low		90	nsec	
	$t_{DL\overline{\Phi}}$ (IR)	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ Low		110	nsec	
	$t_{DH\Phi}$ (IR)	$\overline{IORQ}$ Delay From Rising Edge of Clock, $\overline{IORQ}$ High		100	nsec	C_L = 50pF
	$t_{DH\overline{\Phi}}$ (IR)	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ High		110	nsec	
$\overline{RD}$	$t_{DL\Phi}$ (RD)	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ Low		100	nsec	
	$t_{DL\overline{\Phi}}$ (RD)	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ Low		130	nsec	
	$t_{DH\Phi}$ (RD)	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ High		100	nsec	C_L = 50pF
	$t_{DH\overline{\Phi}}$ (RD)	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ High		110	nsec	
$\overline{WR}$	$t_{DL\Phi}$ (WR)	$\overline{WR}$ Delay From Rising Edge of Clock, $\overline{WR}$ Low		80	nsec	
	$t_{DL\overline{\Phi}}$ (WR)	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ Low		90	nsec	
	$t_{DH\overline{\Phi}}$ (WR)	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ High		100	nsec	C_L = 50pF
	t_{w} ($\overline{WRL}$)	Pulse Width $\overline{WR}$ Low	[10]		nsec	
$\overline{M1}$	t_{DL} (M1)	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ Low		130	nsec	
	t_{DH} (M1)	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ High		130	nsec	C_L = 50pF
$\overline{RFSH}$	t_{DL} (RF)	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ Low		180	nsec	
	t_{DH} (RF)	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ High		150	nsec	C_L = 50pF
$\overline{WAIT}$	t_{S} (WT)	$\overline{WAIT}$ Setup Time to Falling Edge of Clock	70		nsec	
$\overline{HALT}$	t_{D} (HT)	$\overline{HALT}$ Delay Time From Falling Edge of Clock		300	nsec	C_L = 50pF
$\overline{INT}$	t_{S} (IT)	$\overline{INT}$ Setup Time to Rising Edge of Clock	80		nsec	
$\overline{NMI}$	t_{w} ($\overline{NML}$)	Pulse Width, $\overline{NMI}$ Low	80		nsec	
$\overline{BUSRQ}$	t_{S} (BQ)	$\overline{BUSRQ}$ Setup Time to Rising Edge of Clock	80		nsec	
$\overline{BUSAK}$	t_{DL} (BA)	$\overline{BUSAK}$ Delay From Rising Edge of Clock, $\overline{BUSAK}$ Low		120	nsec	
	t_{DH} (BA)	$\overline{BUSAK}$ Delay From Falling Edge of Clock, $\overline{BUSAK}$ High		110	nsec	C_L = 50pF
$\overline{RFSFT}$	t_{S} (RS)	$\overline{RFSET}$ Setup Time to Rising Edge of Clock	90		nsec	
	t_F (C)	Delay to Float ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ and $\overline{WR}$)		100	nsec	
	t_{mr}	$\overline{M1}$ Stable Prior to $\overline{IORQ}$ (Interrupt Ack)	[11]		nsec	

[12] $t_c = t_{w(\Phi H)} + t_{w(\Phi L)} + t_r + t_f$

[1] $t_{acm} = t_{w(\Phi H)} + t_f - 75$ 75

[2] $t_{aci} = t_c - 80$

[3] $t_{ca} = t_{w(\Phi L)} + t_r - 40$

[4] $t_{caf} = t_{w(\Phi L)} + t_r - 60$

[5] $t_{dcm} = t_c - 210$

[6] $t_{dci} = t_{w(\Phi L)} + t_r - 210$

[7] $t_{cdf} = t_{w(\Phi L)} + t_r - 80$

[8] $t_{w(MRL)} = t_c - 40$

[9] $t_{w(MRH)} = t_{w(\Phi H)} + t_f - - 30$

[10] $t_{w(\overline{WR}L)} = t_c - 40$

[11] $t_{mr} = 2t_c + t_{w(\Phi H)} + t_f - 80$

NOTES

A Data should be enabled onto the CPU data bus when $\overline{RD}$ is active. During interrupt acknowledge data should be enabled when $\overline{M1}$ and $\overline{IORQ}$ are both active.

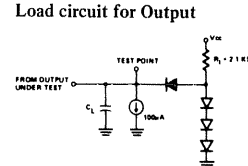
B All control signals are internally synchronized so they may be totally asynchronous with respect to the clock.

C The $\overline{RESET}$ signal must be active for a minimum of 3 clock cycles.

D Output Delay vs. Loaded Capacitance
TA = 70°C Vcc = +5V ±5%
Add 10nsec delay for each 50pf increase in load up to a maximum of 200pf for the data bus & 100pf for address & control lines

E Although static by design, testing guarantees $t_{w(\Phi H)}$ of 200 μsec maximum.

Load circuit for Output



Z80A-CPU A.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= +5V ± 5%, Unless Otherwise Noted.

Signal	Symbol	Parameter	Min	Max	Unit	Test Condition
Φ	t_c	Clock Period	.25	[12]	μsec	
	t_w (ΦH)	Clock Pulse Width, Clock High	110	[E]	nsec	
	t_w (ΦL)	Clock Pulse Width, Clock Low	110	2000	nsec	
	$t_{r,f}$	Clock Rise and Fall Time		30	nsec	
A_{0-15}	t_{D} (AD)	Address Output Delay		110	nsec	C_L = 50pF
	t_{F} (AD)	Delay to Float		90	nsec	
	t_{acm}	Address Stable Prior to MREQ (Memory Cycle)	[1]		nsec	
	t_{aci}	Address Stable Prior to IORQ, RD or WR (I/O Cycle)	[2]		nsec	
	t_{ca}	Address Stable from RD, WR, IORQ or MREQ	[3]		nsec	
	t_{caf}	Address Stable From RD or WR During Float	[4]		nsec	
D_{0-7}	t_{D} (D)	Data Output Delay		150	nsec	C_L = 50pf
	t_{F} (D)	Delay to Float During Write Cycle		90	nsec	
	$t_{S\Phi}$ (D)	Data Setup Time to Rising Edge of Clock During M1 Cycle	35		nsec	
	$t_{S\overline{\Phi}}$ (D)	Data Setup Time to Falling Edge of Clock During M2 to M5	50		nsec	
	t_{dcm}	Data Stable Prior to WR (Memory Cycle)	[5]		nsec	
	t_{dci}	Data Stable Prior to WR (I/O Cycle)	[6]		nsec	
	t_{cdf}	Data Stable From WR	[7]			
	t_H	Any Hold Time for Setup Time		0	nsec	
MREQ	$t_{DL\overline{\Phi}}$ (MR)	MREQ Delay From Falling Edge of Clock MREQ Low		85	nsec	C_L = 50pF
	$t_{DH\Phi}$ (MR)	MREQ Delay From Rising Edge of Clock, MREQ High		85	nsec	
	$t_{DH\overline{\Phi}}$ (MR)	MREQ Delay From Falling Edge of Clock, MREQ High		85	nsec	
	t_w (MRL)	Pulse Width, MREQ Low	[8]		nsec	
	t_w (MRH)	Pulse Width, MREQ High	[9]		nsec	
IORQ	$t_{DL\Phi}$ (IR)	IORQ Delay From Rising Edge of Clock IORQ Low		75	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}}$ (IR)	IORQ Delay From Falling Edge of Clock, IORQ Low		85	nsec	
	$t_{DH\Phi}$ (IR)	IORQ Delay From Rising Edge of Clock, IORQ High		85	nsec	
	$t_{DH\overline{\Phi}}$ (IR)	IORQ Delay From Falling Edge of Clock IORQ High		85	nsec	
RD	$t_{DL\Phi}$ (RD)	RD Delay From Rising Edge of Clock, RD Low		85	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}}$ (RD)	RD Delay From Falling Edge of Clock, RD Low		95	nsec	
	$t_{DH\Phi}$ (RD)	RD Delay From Rising Edge of Clock RD High		85	nsec	
	$t_{DH\overline{\Phi}}$ (RD)	RD Delay From Falling Edge of Clock RD High		85	nsec	
WR	$t_{DL\Phi}$ (WR)	WR Delay From Rising Edge of Clock, WR Low		65	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}}$ (WR)	WR Delay From Falling Edge of Clock, WR Low		80	nsec	
	$t_{DH\overline{\Phi}}$ (WR)	WR Delay From Falling Edge of Clock, WR High		80	nsec	
	t_w (WRL)	Pulse Width, WR Low	[10]		nsec	
M1	t_{DL} (M1)	M1 Delay From Rising Edge of Clock M1 Low		100	nsec	C_L = 50pF
	t_{DH} (M1)	M1 Delay From Rising Edge of Clock M1 High		100	nsec	
RFSH	t_{DL} (RF)	RFSH Delay From Rising Edge of Clock RFSH Low		130	nsec	C_L = 50pF
	t_{DH} (RF)	RFSH Delay From Rising Edge of Clock RFSH High		120	nsec	
WAIT	t_s (WT)	WAIT Setup Time to Falling Edge of Clock	70		nsec	
HALT	t_D (HT)	HALT Delay Time From Falling Edge of Clock		300	nsec	C_L = 50pF
INT	t_s (IT)	INT Setup Time to Rising Edge of Clock	80		nsec	
NMI	t_w (NML)	Pulse Width NMI Low	80		nsec	
BUSRQ	t_s (BQ)	BUSRQ Setup Time to Rising Edge of Clock	50		nsec	
BUSAK	t_{DL} (BA)	BUSAK Delay From Rising Edge of Clock, BUSAK Low		100	nsec	C_L = 50pF
	t_{DH} (BA)	BUSAK Delay From Falling Edge of Clock, BUSAK High		100	nsec	
RESET	t_s (RS)	RESET Setup Time to Rising Edge of Clock	60		nsec	
	t_F (C)	Delay to Float (MREQ, IORQ RD and WR)		80	nsec	
	t_{mr}	M1 Stable Prior to IORQ (Interrupt Ack)	[11]		nsec	

[12] $t_c = t_{w(\Phi H)} + t_{w(\Phi L)} + t_r + t_f$

[1] $t_{acm} = t_{w(\Phi H)} + t_f - 65$

[2] $t_{aci} = t_c - 70$

[3] $t_{ca} = t_{w(\Phi L)} + t_r - 50$

[4] $t_{caf} = t_{w(\Phi L)} + t_r - 45$

[5] $t_{dcm} = t_c - 170$

[6] $t_{dci} = t_{w(\Phi L)} + t_f - 170$

[7] $t_{cdf} = t_{w(\Phi L)} + t_r - 70$

[8] $t_{w(\overline{MRL})} = t_c - 30$

[9] $t_{w(\overline{MRH})} = t_{w(\Phi H)} + t_f - 20$

[10] $t_{w(\overline{WRL})} = t_c - 30$

[11] $t_{mr} = 2t_c + t_{w(\Phi H)} + t_f - 65$

NOTES:

A Data should be enabled onto the CPU data bus when RD is active. During interrupt acknowledge data should be enabled when M1 and IORQ are both active.

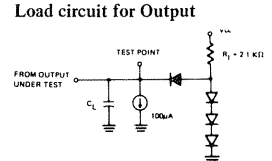
B All control signals are internally synchronized so they may be totally asynchronous with respect to the clock.

C The RESET signal must be active for a minimum of 3 clock cycles.

D Output Delay vs. Loaded Capacitance
TA = 70°C Vcc = +5V ±5%
Add 10nsec delay for each 50pf increase in load up to maximum of 200pf for data bus and 100pf for address & control lines.

E Although static by design, testing guarantees $t_{w(\Phi H)}$ of 200 μsec maximum.

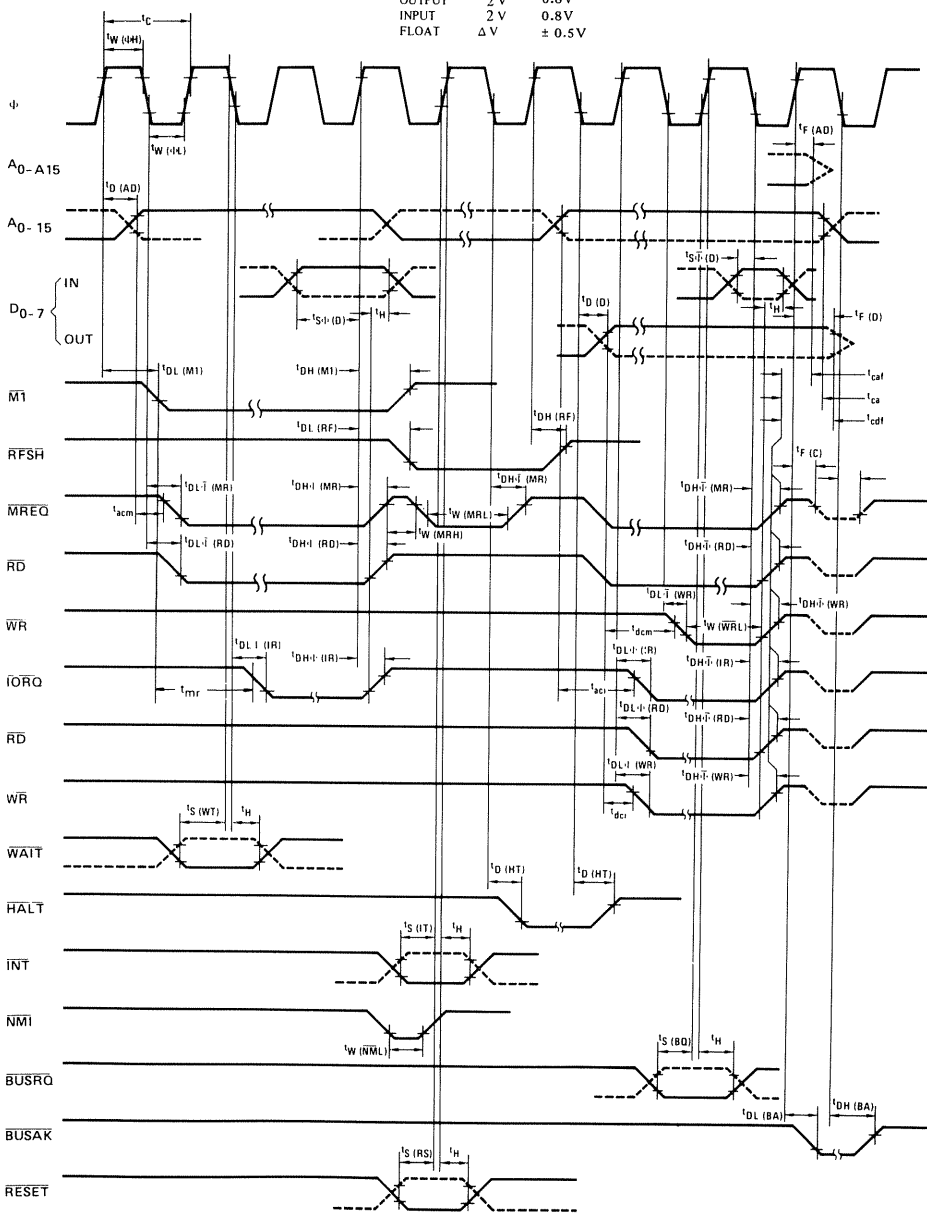
Load circuit for Output



A.C. Timing Diagram

Timing measurements are made at the following voltages, unless otherwise specified.

	"1"	"0"
CLOCK	V_{cc}-0.6V	0.45V
OUTPUT	2V	0.8V
INPUT	2V	0.8V
FLOAT	ΔV	± 0.5V



Absolute Maximum Ratings

Temperature Under Bias	Specified operating range
Storage Temperature	-65°C to +150° C
Voltage On Any Pin with Respect to Ground	-0.3V to +7V
Power Dissipation	1.5W

Note: For Z80-CPU all AC and DC characteristics remain the same for the military grade parts except I_{CC}.
I_{CC} = 200 mA

* Comment
Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Z80-CPU D.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= 5V ± 5% unless otherwise specified

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3		0.45	V	
V_{IHC}	Clock Input High Voltage	V_{CC}-0.6		V_{CC}+0.3	V	
V_{IL}	Input Low Voltage	-0.3		0.8	V	
V_{IH}	Input High Voltage	2		V_{CC}	V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL}= 1.8 mA
V_{OH}	Output High Voltage	2.4			V	I_{OH}= -250 μA
I_{CC}	Power Supply Current			150	mA	
I_{LI}	Input Leakage Current			10	μA	V_{IN}= 0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float			10	μA	V_{OUT}= 2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float			-10	μA	V_{OUT}= 0.4V
I_{LD}	Data Bus Leakage Current in Input Mode			±10	μA	$0 \leq V_{IN} \leq V_{CC}$

Capacitance

T_A= 25°C, f = 1 MHz,
unmeasured pins returned to ground

Symbol	Parameter	Max.	Unit
C_Φ	Clock Capacitance	35	pF
C_{IN}	Input Capacitance	5	pF
C_{OUT}	Output Capacitance	10	pF

Z80A-CPU D.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= 5V ± 5% unless otherwise specified

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3		0.45	V	
V_{IHC}	Clock Input High Voltage	V_{CC}-0.6		V_{CC}+0.3	V	
V_{IL}	Input Low Voltage	-0,3		0.8	V	
V_{IH}	Input High Voltage	2		V_{CC}	V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL}= 1.8 mA
V_{OH}	Output High Voltage	2.4			V	I_{OH}= -250 μA
I_{CC}	Power Supply Current		90	200	mA	
I_{LI}	Input Leakage Current			10	μA	V_{IN}= 0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float			10	μA	V_{OUT}= 2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float			-10	μA	V_{OUT}= 0.4V
I_{LD}	Data Bus Leakage Current in Input Mode			±10	μA	$0 \leq V_{IN} \leq V_{CC}$

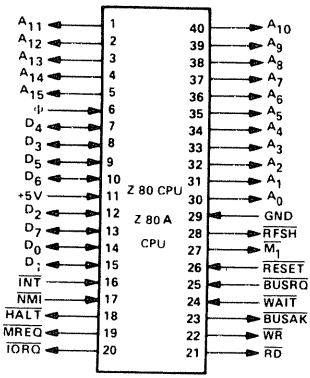
Capacitance

T_A= 25°C, f = 1 MHz,
unmeasured pins returned to ground

Symbol	Parameter	Max.	Unit
C_Φ	Clock Capacitance	35	pF
C_{IN}	Input Capacitance	5	pF
C_{OUT}	Output Capacitance	10	pF

Z 80-CPU
Z 80A-CPU

PIN CONNECTIONS

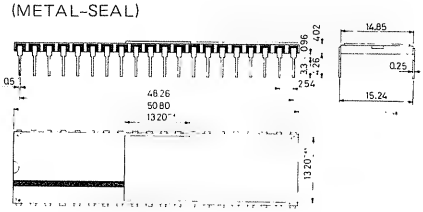


ORDERING NUMBERS:

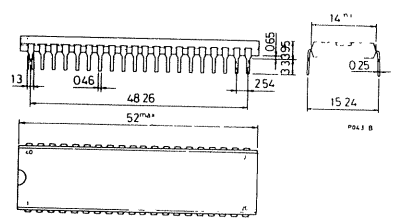
Z80CPU	D1	for dual in-line ceramic package (metal-seal)
Z80CPU	B1	for dual in-line plastic package
Z80ACPU	D1	for dual in-line ceramic package (metal-seal)
Z80ACPU	B1	for dual in-line plastic package
Z80CPU	F1	for dual in-line ceramic package (frit-seal)
Z80ACPU	F1	for dual in-line ceramic package (frit-seal)

MECHANICAL DATA (dimensions in mm)

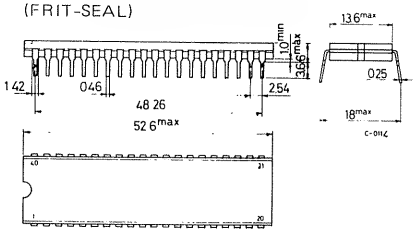
40-PIN CERAMIC DUAL IN-LINE PACKAGE (METAL-SEAL)



40-PIN PLASTIC DUAL IN-LINE PACKAGE



40-PIN CERAMIC DUAL IN-LINE PACKAGE (FRIT-SEAL)



Z 80-PIO
Z 80A-PIO

Product Specification

The SGS-ATES Z80 product line is a complete set of microcomputer components, development systems and support software. The Z-80 microcomputer component set includes all of the circuits necessary to build high-performance microcomputer systems with virtually no other logic and a minimum number of low cost standard memory elements.

The Z-80 Parallel I/O (PIO) Interface Controller is a programmable, two port device which provides TTL compatible interfacing between peripheral devices and the Z80-CPU. The Z80-CPU configures the Z80-PIO to interface with standard peripheral devices such as tape punches, printers, keyboards, etc.

Structure

- N-Channel Silicon Gate Depletion Load technology
- 40 Pin DIP
- Single 5 volt supply
- Single phase 5 volt clock
- Two independent 8-bit bidirectional peripheral interface ports with "handshake" data transfer control

Features

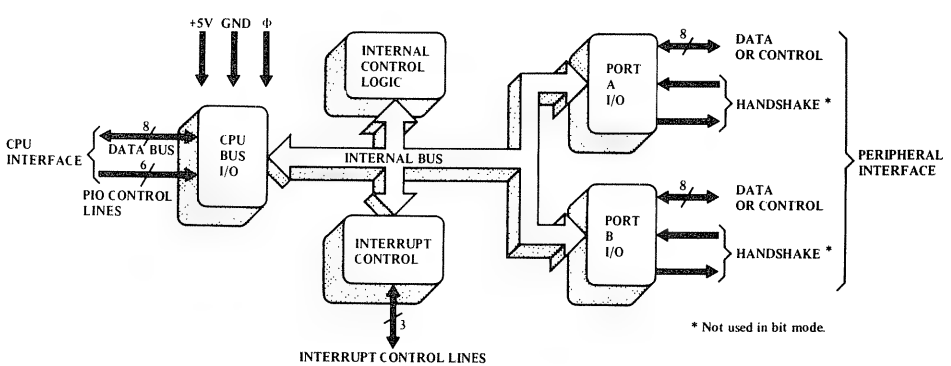
- Interrupt driven "handshake" for fast response
- Any one of the following modes of operation may be selected for either port:
  - Byte output
  - Byte input
  - Byte bidirectional bus (available on Port A only)
  - Bit Mode
- Programmable interrupts on peripheral status conditions
- Daisy chain priority interrupt logic included to provide for automatic interrupt vectoring without external logic.
- Eight outputs are capable of driving Darlington transistors.
- All inputs and outputs fully TTL compatible.

PIO Architecture

A block diagram of the Z80-PIO is shown in figure 3. The internal structure of the Z80-PIO consists of a Z80-CPU bus interface, internal control logic, Port A I/O logic, Port B I/O logic, and interrupt control logic. A typical application might use Port A as the data transfer channel and Port B for the status and control monitoring.

The Port I/O logic is composed of 6 registers with "handshake" control logic as shown in figure 4. The registers include: an 8-bit input register, an 8-bit output register, a 2-bit mode control register, an 8-bit mask register, an 8-bit input/output select register, and a 2-bit mask control register. The last three registers are used only when the port has been programmed to operate in the bit mode.

Fig. 3 - PIO BLOCK DIAGRAM



Register Description

Mode Control Register–2 bits, loaded by CPU to select the operating mode: byte output, byte input, byte bidirectional bus or bit mode.

Data Output Register–8 bits, permits data to be transferred from the CPU to the peripheral.

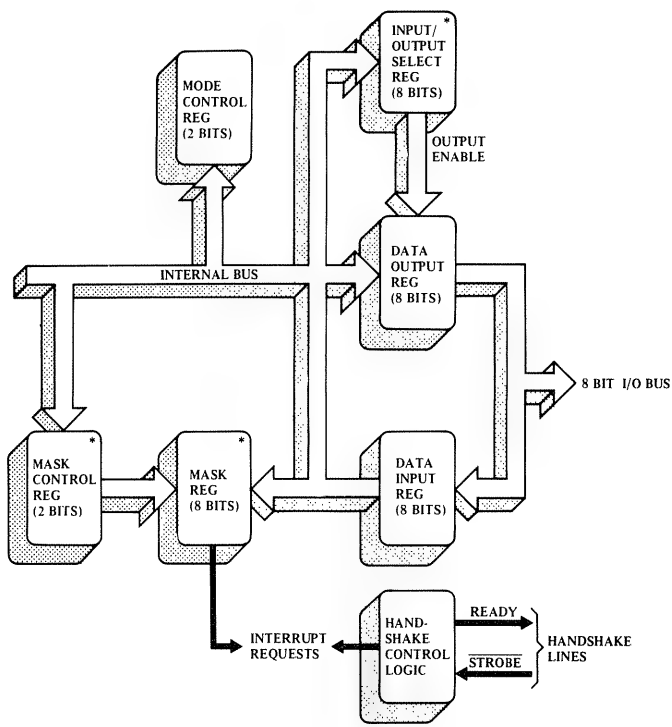
Data Input Register–8 bits, accepts data from the peripheral for transfer to the CPU.

Mask Control Register–2 bits, loaded by the CPU to specify the active state (high or low) of any peripheral device interface pins that are to be monitored and, if an interrupt should be generated when all unmasked pins are active (AND condition) or, when any unmasked pin is active (OR condition).

Mask Register–8 bits, loaded by the CPU to determine which peripheral device interface pins are to be monitored for the specified status condition.

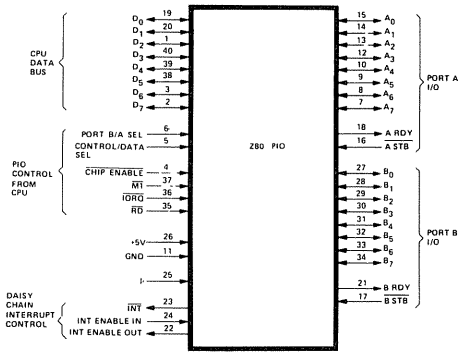
Input/Output Select Register–8 bits, loaded by the CPU to allow any pin to be an output or an input during bit mode operation.

Fig. 4 – A TYPICAL PORT I/O BLOCK DIAGRAM



* Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state

Z80-PIO Pin Description



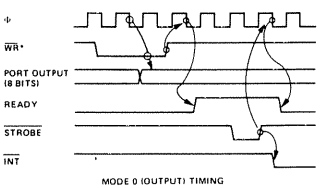
D_7-D_0	Z80-CPU Data Bus (bidirectional, tristate)
B/A Sel	Port B or A Select (input, active high)
C/D Sel	Control or Data Select (input, active high)
$\overline{CE}$	Chip Enable (input, active low)
Φ	System Clock (input)

$\overline{M1}$	Machine Cycle One Signal from CPU (input, active low)
$\overline{IORQ}$	Input/Output Request from Z80-CPU (input, active low)
$\overline{RD}$	Read Cycle Status from the Z80-CPU (input, active low)
IEI	Interrupt Enable In (input, active high)
IEO	Interrupt Enable Out (output, active high). IEI and IEO form a daisy chain connection for priority interrupt control
$\overline{INT}$	Interrupt Request (output, open drain, active low)
A_0-A_7	Port A Bus (bidirectional, tristate)
$\overline{A\ STB}$	Port A Strobe Pulse from Peripheral Device (input, active low)
A RDY	Register A Ready (output, active high)
B_0-B_7	Port B Bus (bidirectional, tristate)
$\overline{B\ STB}$	Port B Strobe Pulse from Peripheral Device (input, active low)
B RDY	Register B Ready (output, active high)

Timing Waveforms

OUTPUT MODE

An output cycle is always started by the execution of an output instruction by the CPU. The $\overline{WR}$ pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The write pulse sets the ready flag after a low going edge of Φ, indicating data is available. Ready stays active until the positive edge of the strobe line is received indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an $\overline{INT}$ if the interrupt enable flip flop has been set and if this device has the highest priority.

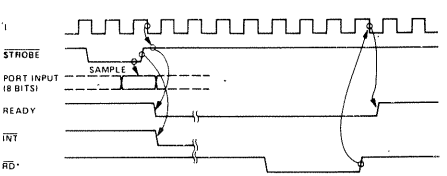


MODE 0 (OUTPUT) TIMING

WR* = $\overline{RD}$ CE $\overline{C/D}$ IORQ

INPUT MODE

When $\overline{STROBE}$ goes low data is loaded into the selected port input register. The next rising edge of strobe activates $\overline{INT}$ if interrupt enable is set and this is the highest priority requesting device. The following falling edge of Φ resets Ready to an inactive state, indicating that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete the positive edge of $\overline{RD}$ will set Ready at the next low going transition of Φ. At this time new data can be loaded into the PIO.

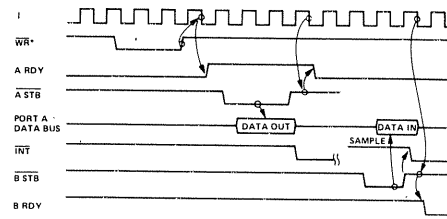


RD* = RD CE $\overline{C/D}$ IORQ MODE 1 (INPUT) TIMING

Timing Waveforms (continued)

BIDIRECTIONAL MODE

This is a combination of modes 0 and 1 using all four handshake lines and the 8 Port A I/O lines. Port B must be set to the Bit Mode. The Port A handshake lines are used for output control and the Port B lines are used for input control. Data is allowed out onto the Port A bus only when $\overline{A\ STB}$ is low. The rising edge of this strobe can be used to latch the data into the peripheral.

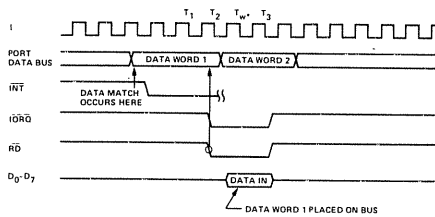


$\overline{WR}^* = \overline{RD} \cdot CE \cdot \overline{C/D} \cdot \overline{IORQ}$

BIT MODE

The bit mode does not utilize the handshake signals and a normal port write or port read can be executed at any time. When writing, the data will be latched into the output registers with the same timing as the output mode.

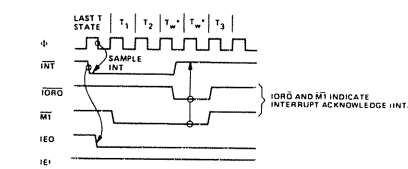
When reading the PIO, the data returned to the CPU will be composed of output register data from those port data lines assigned as outputs and input register data from those port data lines assigned as inputs. The input register will contain data which was present immediately prior to the falling edge of $\overline{RD}$. An interrupt will be generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers



* Timing Diagram Refers to Bit Mode Read

INTERRUPT ACKNOWLEDGE

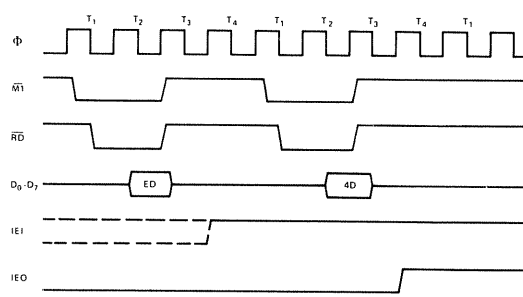
During $\overline{M1}$ time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the $\overline{INT}$ Enable signal to ripple through the daisy chain. The peripheral with IEI high and IEO low during $\overline{INTA}$ will place a preprogrammed 8-bit interrupt vector on the data bus at this time. IEO is held low until a return from interrupt (RETI) instruction is executed by the CPU while IEI is high. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.



RETURN FROM INTERRUPT CYCLE

If a Z80 peripheral device has no interrupt pending and is not under service, then its IEO=IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always low, inhibiting lower priority chips from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO will be low unless an "ED" is decoded as the first byte of a two byte opcode. In this case, IEO will go high until the next opcode byte is decoded, whereupon it will again go low. If the second byte of the opcode was a "4D" then the opcode was an RETI instruction.

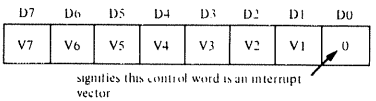
After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service will have its IEI high and its IEO low. This device is the highest priority device in the daisy chain which has received an interrupt acknowledge. All other peripherals have IEI=IEO. If the next opcode byte decoded is "4D", this peripheral device will reset its "interrupt under service" condition.



PIO Programming

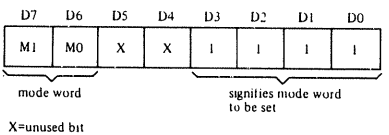
LOAD INTERRUPT VECTOR

The Z80-CPU requires an 8-bit interrupt vector be supplied by the interrupting device. The CPU forms the address for the interrupt service routine of the port using this vector. During an interrupt acknowledge cycle the vector is placed on the Z-80 data bus by the highest priority device requesting service at that time. The desired interrupt vector is loaded into the PIO by writing a control word to the desired port of the PIO with the following format.



SELECTING AN OPERATING MODE

When selecting an operating mode, the 2-bit mode control register is set to one of four values. These two bits are the most significant bits of the register, bits 7 and 6; bits 5 and 4 are not used while bits 3 through 0 are all set to 1111 to indicate "set mode."



Mode	M_1	M_0
Output	0	0
Input	0	1
Bidirectional	1	0
Bit	1	1

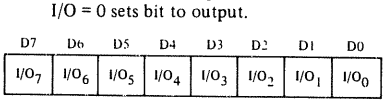
MODE 0 active indicates that data is to be written from the CPU to the peripheral.

MODE 1 active indicates that data is to be read from the peripheral to the CPU.

MODE 2 allows data to be written to or read from the peripheral device.

MODE 3 is intended for status and control applications. When selected, the next control word must set the I/O Register to indicate which lines are to be input and which lines are to be output.

I/O = 1 sets bit to input.
I/O = 0 sets bit to output.



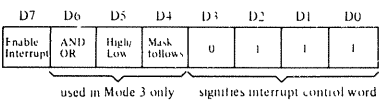
INTERRUPT CONTROL

Bit 7 = 1 — interrupt enable is set–allowing interrupt to be generated.

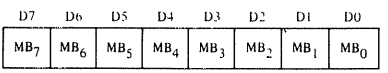
Bit 7 = 0 — indicates the enable flag is reset and interrupts may not be generated.

Bits 6,5,4 — are used in the bit mode interrupt operations; otherwise they are disregarded.

Bits 3,2,1,0 — signify that this command word is an interrupt control word.

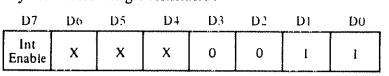


If the "mask follows" bit is high (D4 = 1), the next control word written to the port must be the mask.



Only those port lines whose mask bit is a 0 will be monitored for generating an interrupt

The interrupt enable flip-flop of a port may be set or reset without modifying the rest of the interrupt control word by the following command.



Z80-PIO A.C. Characteristics

TA = 0° C to 70° C, Vcc = +5 V ± 5%, unless otherwise noted

Number	Symbol	Parameter	Min	Max	Unit	Comments
1	TcC	Clock Cycle Time	250	[1]	ns	
2	TcCh	Clock Width (High)	105	2000	ns	
3	TcCL	Clock Width (Low)	105	2000	ns	
4	TfC	Clock Fall Time		30	ns	
5	TrC	Clock Rise Time		30	ns	
6	TsCS(RI)	$\overline{CE}$, B/$\overline{A}$, C/$\overline{D}$ to $\overline{RD}$, $\overline{IORQ}$ ↓ Setup Time	50		ns	[6]
7	Th	Any Hold Time for specified Setup Time	0		ns	
8	TsRI(C)	$\overline{RD}$, $\overline{IORQ}$ to Clock ↑ Setup Time	115		ns	
9	TdRI(DO)	$\overline{RD}$, $\overline{IORQ}$ ↓ to Data Out Delay		380	ns	[2]
10	TdRI(DOr)	$\overline{RD}$, $\overline{IORQ}$ ↑ to Data Out Float Delay		110	ns	
11	TsDI(C)	Data In to Clock ↑ Setup Time	50		ns	CL = 50 pF
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)	250		ns	[3]
13	TsM1(Cr)	$\overline{M1}$ ↓ to Clock ↑ Setup Time	90		ns	
14	TsM1(Cf)	$\overline{M1}$ ↑ to Clock ↓ Setup Time (M1 Cycle)	0		ns	
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt immediately preceding M1)		190	ns	[5] See Note A
16	TsIEI(IO)	IEI to IORQ ↓ Setup Time (INTA Cycle)	140		ns	See Note A
17	TdIEI(IEO)	IEI ↓ to IEO ↓ Delay		130	ns	[5] CL = 50 pF
18	TdIEI(IOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)		160	ns	[5]
19	TsIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	220		ns	
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay	200		ns	[5] CL = 50 pF
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay	150		ns	[5]
22	TwSTB	$\overline{STROBE}$ Pulse Width	150 [4]		ns	
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (To Activate $\overline{READY}$ on Next Clock Cycle)	200		ns	
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT data stable Delay (Mode 0)		180	ns	[5]
25	TsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (Mode 1)	230		ns	
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (Mode 2)		210	ns	[5]
27	TdSTB(PDz)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (Mode 2)		180	ns	CL = 50 pF
28	TdPD(INT)	PORT DATA Match to INT ↓ Delay (Mode 3)		490	ns	
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to $\overline{INT}$ ↓ Delay		440	ns	

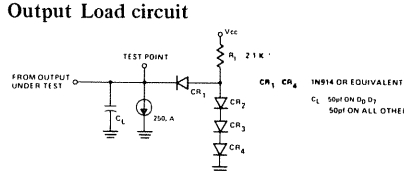
Notes A 2 5 TcC > (N-2) TdIEI (IEOG) + TdM1(IEO) + TsIEI(IO)+TTL Buffer Delay. if any
B. M1 Must be active for a minimum of 2 clock cycles to reset the PIO
[1] TcC = TwCh + TwCl + TrC + TfC
[2] Increase TdRI(DO) by 10 nsec for each 50 pF increase in loading up to 200 pF max
[3] Increase TdIO(DOT) by 10 nsec for each 60 pF increase in loading up to 200 pF max
[4] For Mode 2 TwSTB > TsPD(STB)
[5] Increase these values by 2 nsec for each 10 pF increase in loading up to 100 pF max
[6] TsCS(RI) may be reduced However the time subtracted from TsCS(RI) will be added to TdRI(DO)

Capacitance

TA = 25° C, f = 1 MHz

Symbol	Parameter	Max	Unit	Test Condition
C_Φ	Clock Capacitance	10	pF	Unmeasured Pins Returned to Ground
C_{IN}	Input Capacitance	5	pF	
C_{OUT}	Output Capacitance	10	pF	

Output Load circuit



Z80A-PIO A.C. Characteristics

TA = 0° C to 70° C, Vcc = +5 V ± 5%, unless otherwise noted

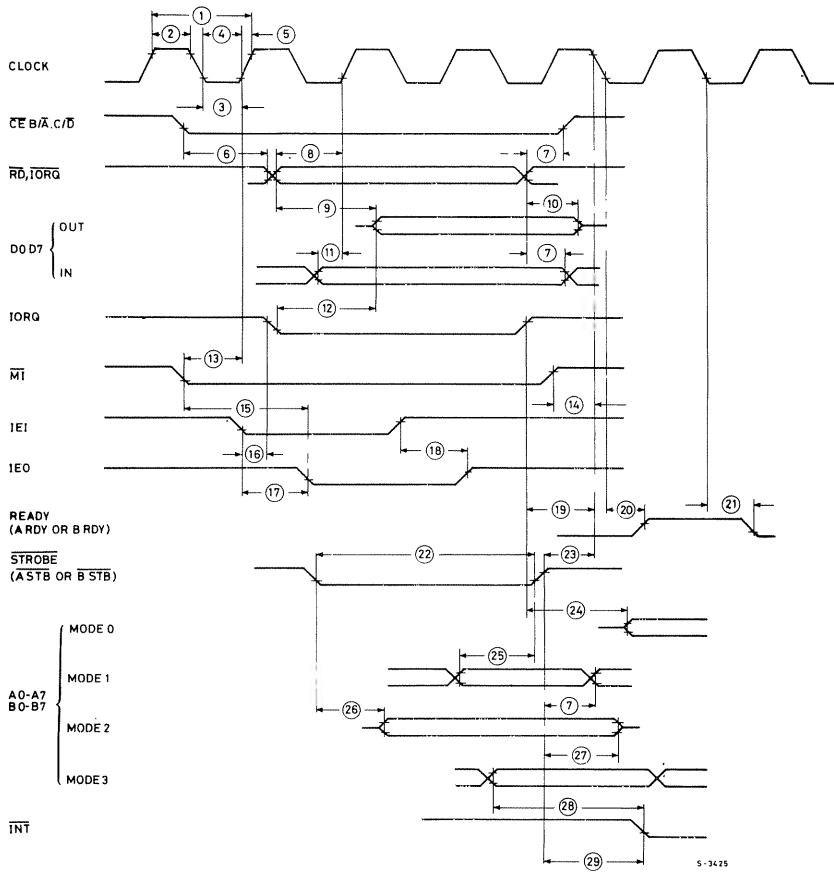
Number	Symbol	Parameter	Min	Max	Unit	Comments
1	TcC	Clock Cycle Time	400	[1]	ns	
2	TcCh	Clock Width (High)	170	2000	ns	
3	TcCL	Clock Width (Low)	170	2000	ns	
4	TfC	Clock Fall Time		30	ns	
5	TrC	Clock Rise Time		30	ns	
6	TsCS(RI)	$\overline{CE}$ B/$\overline{A}$, C/$\overline{D}$ to $\overline{RD}$, $\overline{IORQ}$ ↓ Setup Time	50		ns	[6]
7	Th	Any Hold Time for specified Setup Time	0		ns	
8	TsRI(C)	$\overline{RD}$, $\overline{IORQ}$ to Clock ↑ Setup Time	115		ns	
9	TdRI(DO)	$\overline{RD}$ $\overline{IORQ}$ ↓ to Data Out Delay		430	ns	[2]
10	TdRI(DOr)	$\overline{RD}$ $\overline{IORQ}$ ↑ to Data Out Float Delay		160	ns	
11	TsDI(C)	Data In to Clock ↑ Setup Time	50		ns	CL = 50 pF
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)	340		ns	[3]
13	TsM1(Cr)	$\overline{M1}$ ↓ to Clock ↑ Setup Time	210		ns	
14	TsM1(Cf)	$\overline{M1}$ ↑ to Clock ↓ Setup Time ($\overline{M1}$ Cycle)	0		ns	
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{M1}$ ↓)		300	ns	[5] See Note A
16	TsIEI(IO)	IEI to IORQ ↓ Setup Time (INTA Cycle)	140		ns	See Note A
17	TdIEI(IEO)	IEI ↓ to IEO ↓ Delay		190	ns	[5] CL = 50 pF
18	TdIEI(IIOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)		210	ns	[5]
19	TsIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	220		ns	
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay	200		ns	[5] CL = 50 pF
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay	150		ns	[5]
22	TwSTB	$\overline{STROBE}$ Pulse Width	150 [4]		ns	
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	220		ns	
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT data stable Delay (Mode 0)		200	ns	[5]
25	TsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (Mode 1)	260		ns	
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (Mode 2)		230	ns	[5]
27	TdSTB(PDz)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (Mode 2)		200	ns	CL = 50 pF
28	TdPD(INT)	PORT DATA Match to INT ↓ Delay (Mode 3)		540	ns	
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to INT ↓ Delay		490	ns	

Notes
A 2.5 TcC > (N-2) TdIEI (IEOG) + TdM1(IEO) + TsIEI(IO) + TTL Buffer Delay, if any
B M1 Must be active for a minimum of 2 clock cycles to reset the PIO
[1] TcC = TwCh + TwCl + TrC + TfC
[2] Increase TdRI(DO) by 10 nsec for each 50 pF increase in loading up to 200 pF max
[3] Increase TdIO(DOT) by 10 nsec for each 60 pF increase in loading up to 200 pF max
[4] For Mode 2 TwSTB > TsPD(STB)
[5] Increase these values by 2 nsec for each 10 pF increase in loading up to 100 pF max
[6] TsCS(RI) may be reduced However the time subtracted from TsCS(RI) will be added to TdRI(DO)

A.C. Timing Diagram

Timing measurements are made at the following voltages, unless otherwise specified:

	"1"	"0"
CLOCK	V_{CC}-0.6V	0.45V
OUTPUT	2V	0.8V
INPUT	2V	0.8V
FLOAT	ΔV	±0.5V



Absolute Maximum Ratings

Temperature Under Bias	Specified operating range
Storage Temperature	-65°C to +150°C
Voltage On Any Pin with Respect to Ground	-0.3V to +7V
Power Dissipation	0.6W

Note: All AC and DC characteristics remain the same for the military grade parts except I_{cc}.
I_{cc}= 130 mA

* Comment
Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

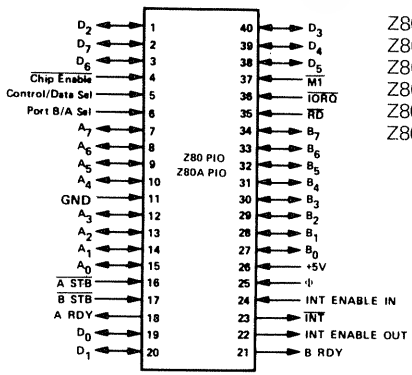
Z80-PIO and Z80A-PIO D.C. Characteristics

T_A= 0°C to 70°C, V_{cc}= +5V ± 5%, unless otherwise noted

Symbol	Parameter	Min.	Max.	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3	0.45	V	
V_{IHC}	Clock Input High Voltage	V_{cc}-0.6	V_{cc}+0.3	V	
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{IH}	Input High Voltage	2	Vcc	V	
V_{OL}	Output Low Voltage		0.4	V	I_{OL} = 2.0 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} - 250 μA
I_{CC}	Power Supply Current		70	mA	
I_{LI}	Input Leakage Current		10	μA	V_{IN} = 0 to Vcc
I_{LOH}	Tri-State Output Leakage Current in Float		10	μA	V_{OUT} = 2.4 to Vcc
I_{LOL}	Tri-State Output Leakage Current in Float		-10	μA	V_{OUT} = 0.4 V
I_{LD}	Data Bus Leakage Current in Input Mode		±10	μA	$0 \leqslant V_{IN} \leqslant Vcc$
I_{OHD}	Darlington Drive Current	-1.5		mA	V_{OH} = 1.5 V Port B Only

Z 80-PIO
Z 80A-PIO

PIN CONNECTIONS

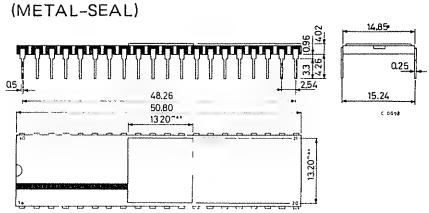


ORDERING NUMBERS:

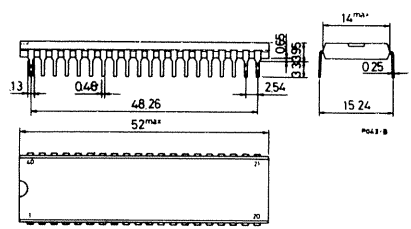
Z80PIO	D1	for dual in-line ceramic package (metal-seal)
Z80PIO	B1	for dual in-line plastic package
Z80APIO	D1	for dual in-line ceramic paçkage (metal-seal)
Z80APIO	B1	for dual in-line plastic package
Z80PIO	F1	for dual in-line ceramic package (frit-seal)
Z80APIO	F1	for dual in-line ceramic package (frit-seal)

MECHANICAL DATA (dimension in mm)

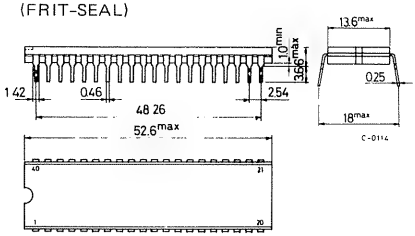
40-PIN CERAMIC DUAL IN-LINE PACKAGE (METAL-SEAL)



40-PIN PLASTIC DUAL IN-LINE PACKAGE



40-PIN CERAMIC DUAL IN-LINE PACKAGE (FRIT-SEAL)



Product Specification

The SGS-ATES Z80 product line is a complete set of microcomputer components, development systems and support software. The Z80 microcomputer component set includes all of the circuits necessary to build high-performance microcomputer systems with virtually no other logic and a minimum number of low cost standard memory elements.

The Z80-Counter Timer Circuit (CTC) is a programmable, four channel device that provides counting and timing functions for the Z80-CPU. The Z80-CPU configures the Z80-CTC's four independent channels to operate under various modes and conditions as required.

Structure

- N-Channel Silicon Gate Depletion Load Technology
- 28 Pin DIP
- Single 5 volt supply
- Single phase 5 volt clock
- Four independent programmable 8-bit counter/16-bit timer channels

Features

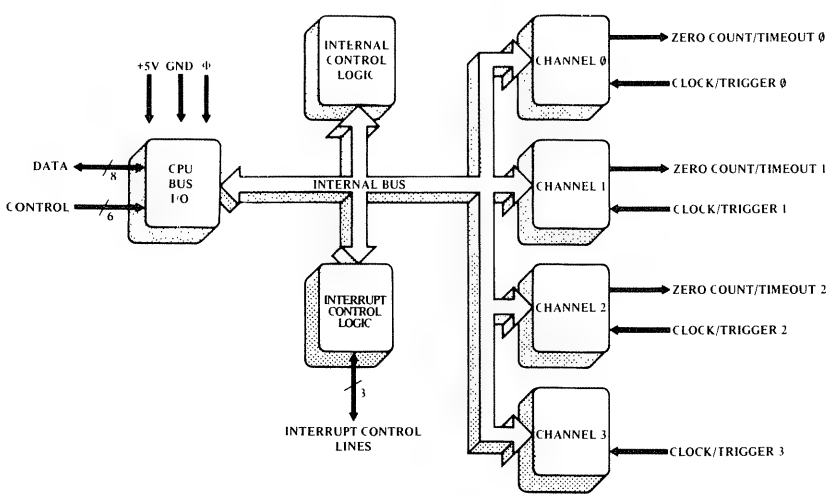
- Each channel may be selected to operate in either a counter mode or timer mode.
- Programmable interrupts on counter or timer states.
- A time constant register automatically reloads the down counter at zero and the cycle is repeated.
- Readable down counter indicates number of counts-to-go until zero.
- Selectable 16 or 256 clock prescaler for each timer channel.
- Selectable positive or negative trigger may initiate timer operation.
- Three channels have zero count/timeout outputs capable of driving Darlington transistors.
- Daisy chain priority interrupt logic included to provide for automatic interrupt vectoring without external logic.
- All inputs and outputs fully TTL compatible.

CTC Architecture

A block diagram of the Z80-CTC is shown in figure 5. The internal structure of the Z80-CTC consists of a Z80-CPU bus interface, internal control logic, four counter channels, and interrupt control logic. Each channel has an interrupt vector for automatic interrupt vectoring, and interrupt priority is determined by channel number with channel Ø having the highest priority.

The channel logic is composed of 2 registers, 2 counters and control logic as shown in figure 6. The registers include an 8-bit time constant register and an 8-bit channel control register. The counters include an 8-bit readable down counter and an 8-bit prescaler. The prescaler may be programmed to divide the system clock by either 16 or 256.

Fig. 5 – CTC BLOCK DIAGRAM



Channel Counter and Register Description

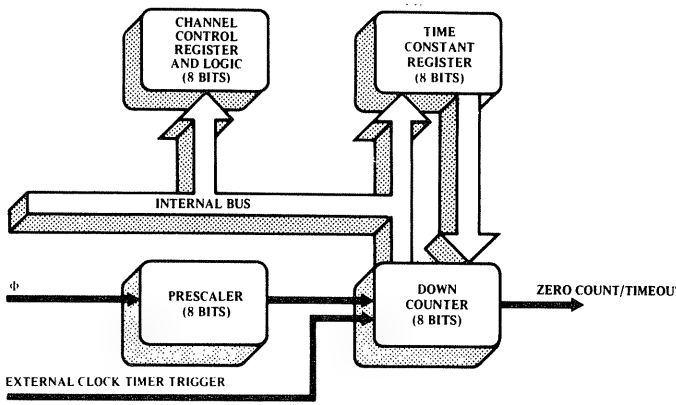
Time Constant Register – 8 bits, loaded by the CPU to initialize and re-load Down Counter at a count of zero.

Channel Control Register – 8 bits, loaded by the CPU to select the mode and conditions of channel operation.

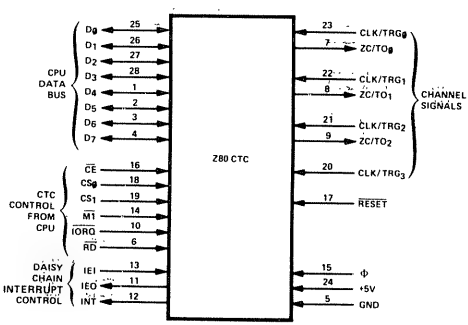
Down Counter – 8 bits, loaded by the Time Constant Register under program control and automatically at a count of zero. At any time, the CPU can read the number of counts-to-go until a zero count. This counter is decremented by the prescaler in timer mode and CLK/TRIG in counter mode.

Prescaler – 8 bit counter, divides system clock by 16 or 256 for decrementing Down Counter. It is used in timer mode only.

Fig. 6 - CHANNEL BLOCK DIAGRAM



Z80-CTC Pin Description



CLK/TRGϕ Channel ϕ External Clock or Timer Trigger (Input)

CLK/TRG1 Channel 1 External Clock or Timer Trigger (Input)

CLK/TRG2 Channel 2 External Clock or Timer Trigger (Input)

CLK/TRG3 Channel 3 External Clock or Timer Trigger (Input)

ZC/TOϕ Channel ϕ Zero Count or Timeout (output, active high)

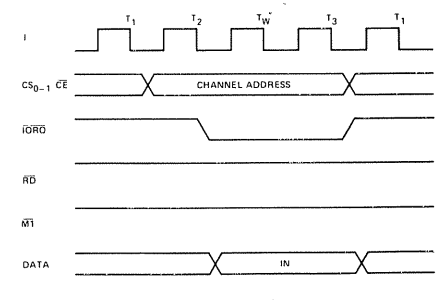
Z80-CTC Pin Description (continued)

ZC/TO_1	Channel 1 Zero Count or Timeout (output, active high)
ZC/TO_2	Channel 2 Zero Count or Timeout (output, active high)
$CS_1 - CS_0$	Channel Select (input, active high). These form a 2-bit binary address of the channel to be accessed.
$D7 - D0$	Z80-CPU Data Bus (bidirectional, tristate)
$\overline{CE}$	Chip Enable (input, active low)
Φ	System Clock (input)
$\overline{M1}$	Machine Cycle One Signal from Z80-CPU (input, active low)
$\overline{IORQ}$	Input/Output Request from Z80-CPU (input, active low)
$\overline{RD}$	Read Cycle Status from the Z80-CPU (input, active low)
IEI	Interrupt Enable In (input, active high)
IEO	Interrupt Enable Out (output, active high). IEI and IEO form a daisy chain connection for priority interrupt control
$\overline{INT}$	Interrupt Request (output, open drain, active low)
$\overline{RESET}$	RESET stops all channels from counting and resets channel interrupt enable bits in all control registers. During reset time ZC/TO_{0-2} and $\overline{INT}$ go to the inactive states, IEO reflects the state of IEI, and the data bus output drivers go to the high impedance state (input, active low)

Timing Waveforms

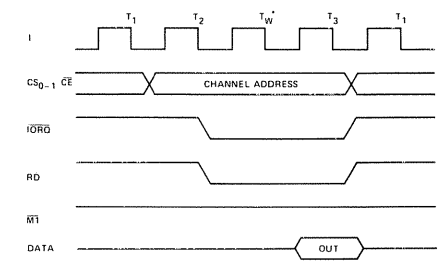
CTC WRITE CYCLE

Illustrated here is the timing for loading a channel control word, time constant and interrupt vector. No wait states are allowed for writing to the CTC other than the automatically inserted (T_W*). Since the CTC does not receive a specific write signal, it internally generates its own from the lack of an $\overline{RD}$ signal.



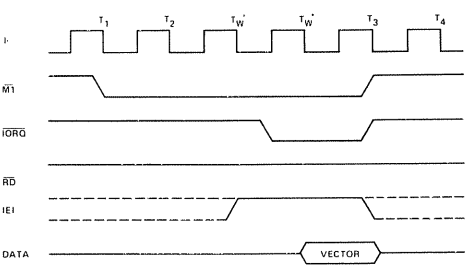
CTC READ CYCLE

Illustrated here is the timing for reading a channel's Down Counter when in Counter Mode. The value read onto the data bus reflects the number of external clock's rising edges prior to the rising edge of cycle (T_2). No wait states are allowed for reading the CTC other than the automatically inserted (T_W*).



INTERRUPT ACKNOWLEDGE CYCLE

Some time after an interrupt is requested by the CTC, the CPU will send out an interrupt acknowledge ($\overline{M1}$ and $\overline{IORQ}$). During this time the interrupt logic of the CTC will determine the highest priority channel which is requesting an interrupt. To insure that the daisy chain enable lines stabilize, channels are inhibited from changing their interrupt request status when $\overline{M1}$ is active. If the CTC Interrupt Enable Input (IEI) is active, then the highest priority interrupting channel places the contents of its interrupt vector register onto the Data Bus when $\overline{IORQ}$ goes active. Additional wait cycles are allowed.



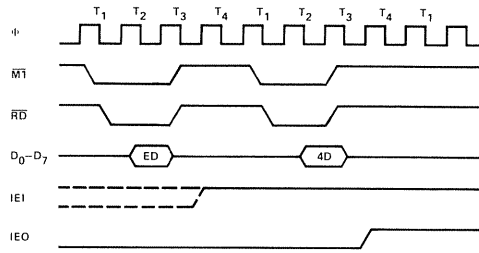
Timing Waveforms (continued)

RETURN FROM INTERRUPT CYCLE

If a Z80 peripheral device has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e. it has already interrupted and received an interrupt acknowledge) then its IEO is always low, inhibiting lower priority chips from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO will be low unless an "ED" is decoded as the first byte of a two byte opcode. In this case, IEO will go high until the next opcode byte is decoded, whereupon it will again go low. If the second byte of the opcode was a "4D" then the opcode was an RETI instruction.

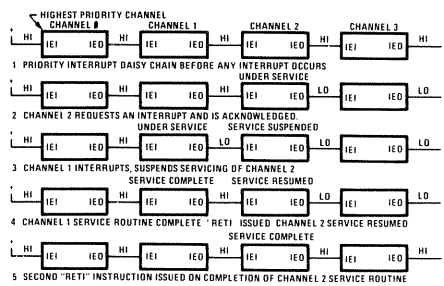
After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service will have its IEI high and its IEO low. This device is the highest priority device in the daisy chain which has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D", this peripheral device will reset its "interrupt under service" condition.

Wait cycles are allowed in the $\overline{M1}$ cycles.



DAISY CHAIN INTERRUPT SERVICING

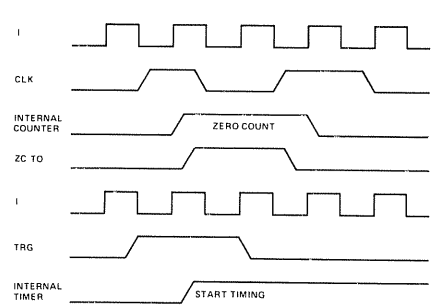
Illustrated at right is a typical nested interrupt sequence which may occur in the CTC. In this sequence channel 2 interrupts and is granted service. While this channel is being serviced, higher priority channel 1 interrupts and is granted service. The service routine for the higher priority channel is completed and a RETI instruction is executed to indicate to the channel that its routine is complete. At this time the service routine of lower priority channel 2 is completed.



CTC COUNTING AND TIMING

In the counter mode the rising or falling edge of the CLK input causes the counter to be decremented. The edge is detected totally asynchronously and must have a minimum CLK pulse width. However, the counter is synchronous with Φ therefore a setup time must be met when it is desired to have the counter decremented by the next rising edge of Φ.

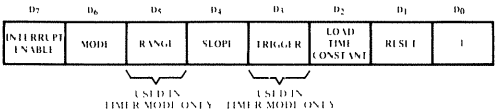
In the timer mode the prescaler may be enabled by a rising or falling edge on the TRG input. As in the counter mode, the edge is detected totally asynchronously and must have a minimum TRG pulse width. However, when timing is to start with respect to the next rising edge of Φ a setup time must be met. The prescaler counts rising edges of Φ.



CTC Programming

SELECTING AN OPERATING MODE

When selecting a channel's operating mode, bit Ø is set to 1 to indicate this word is to be stored in the channel control register.



Bit 7 = Ø — Channel interrupts disabled.

Bit 7 = 1 — Channel interrupts enabled to occur every time Down Counter reaches a count of zero. Setting Bit 7 does not let a preceding count of zero cause an interrupt.

Bit 6 = Ø — Timer Mode – Down counter is clocked by the prescaler. The period of the counter is:
$t_C \bullet P \bullet TC$
t_C = system clock period
P = prescale of 16 or 256
TC = 8 bit binary programmable time constant (256 max)

Bit 6 = 1 — Counter Mode – Down Counter is clocked by external clock. The prescaler is not used.

Bit 5 = Ø — Timer Mode Only–System clock Φ is divided by 16 in prescaler.

Bit 5 = 1 — Timer Mode Only–System clock Φ is divided by 256 in prescaler.

Bit 4 = Ø — Timer Mode – negative edge trigger starts timer operation.
Counter Mode – negative edge decrements the down counter.

Bit 4 = 1 — Timer Mode – positive edge trigger starts timer operation.
Counter Mode – positive edge decrements the down counter.

Bit 3 = Ø — Timer Mode Only – Timer begins operation on the rising edge of T_2 of the machine cycle following the one that loads the time constant.

Bit 3 = 1 — Timer Mode Only – External trigger is valid for starting timer operation after rising edge of T_2 of the machine cycle following the one that loads the time constant. The Prescaler is decremented 2 clock cycles later if the setup time is met, otherwise 3 clock cycles.

Bit 2 = Ø — No time constant will follow the channel control word. One time constant must be written to the channel to initiate operation.

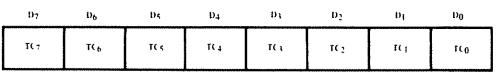
Bit 2 = 1 — The time constant for the Down Counter will be the next word written to the selected channel. If a time constant is loaded while a channel is counting, the present count will be completed before the new time constant is loaded into the Down Counter.

Bit 1 = Ø — Channel continues counting.

Bit 1 = 1 — Stop operation. If Bit 2 = 1 channel will resume operation after loading a time constant, otherwise a new control word must be loaded.

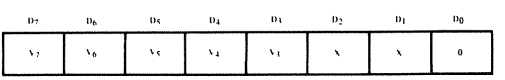
LOADING A TIME CONSTANT

An 8-bit time constant is loaded into the Time Constant register following a channel control word with bit 2 set. All zeros indicate a time constant of 256.



LOADING AN INTERRUPT VECTOR

The Z80-CPU requires that an 8-bit interrupt vector be supplied by the interrupting channel The CPU forms the address for the interrupt service routine of the channel using this vector. During an interrupt acknowledge cycle the vector is placed on the Z80 Data Bus by the highest priority channel requesting service at that time. The desired interrupt vector is loaded into the CTC by writing into channel Ø with a zero in DØ D_7-D_3 contain the stored interrupt vector, D_2 and D_1 are not used in loading the vector. When the CTC responds to an interrupt acknowledge, these two bits contain the binary code of the highest priority channel which requested the interrupt and DØ contains a zero since the address of the interrupt service routine starts at an even byte. Channel Ø is the highest priority channel.



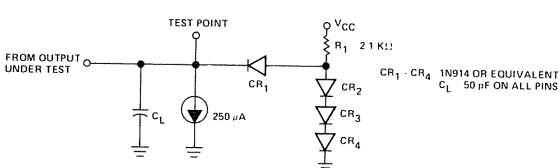
Z80-CTC A.C. Characteristics

TA = 0° C to 70° C, Vcc = +5 V ± 5%, unless otherwise noted

Number	Symbol	Parameter	Min	Max	Unit	Comments
1	TcC	Clock Cycle Time	250	[1]	ns	
2	TwCh	Clock Width (High)	105	2000	ns	
3	TwCL	Clock Width (Low)	105	2000	ns	
4	TfC	Clock Fall Time		30	ns	
5	TrC	Clock Rise Time		30	ns	
6	Th	All Hold Times	0		ns	
* 7	TsCS(C)	CS to Clock ↑ Setup Time	160		ns	
* 8	TsCE(C)	$\overline{CE}$ to Clock ↑ Setup Time	150		ns	
9	TsIO(C)	$\overline{IORQ}$ ↓ to Clock ↑ Setup Time	115		ns	
10	TsRD(C)	$\overline{RD}$ to Clock ↑ Setup Time	115		ns	
*11	TdC(DO)	Clock ↓ to Data Out Delay		200	ns	[2]
12	TdC(DOz)	Clock ↑ to Data Out Float Delay		110	ns	
13	TsDI(C)	Data In to Clock ↑ Setup Time	50		ns	
14	TsM1(C)	$\overline{M1}$ to Clock ↑ Setup Time (INTA or M1 Cycle)	90		ns	
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt immediately preceding M1 ↓) See Note A		190	ns	[3]
16	TdIO(DOT)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)		160	ns	[2]
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		130	ns	[3]
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)		160	ns	[3]
*19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		TcC+160	ns	Timer Mode
*20	TdCTR(INT)	CLK/TRG ↑ to INT ↓				
		TsCTR(C) Satisfied		TcC+160	ns	Counter Mode
		TsCTR(C) not Satisfied		2TcC+370	ns	
21	TcCTR	CLK Cycle Time	2TcC		ns	Counter Mode
22	TrCTR	CLK/TRG Rise Time		50	ns	
23	TfCTR	CLK/TRG Fall Time		50	ns	
24	TwCTRL	CLK/TRG Width (Low)	200		ns	
25	TwCTRh	CLK/TRG Width (High)	200		ns	
26	TsCTR(Cc)	CLK ↑ to Clock ↑ Setup Time for Immediate Count	210		ns	Counter Mode
27	TsCTR(Ct)	TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following Clock ↑	210		ns	Timer Mode
28	TdC(ZCTOr)	Clock ↑ to ZC/TO ↑ Delay	190		ns	
29	TdC(ZCTOf)	Clock ↓ to ZC/TO ↓ Delay	190		ns	

Notes A $2.5\ TcC > (N-2)\ TdIEI(IEOF)+TdM1(IEO)+TsIEI(IO)$
B $\overline{RESET}$ must be active for a minimum of 3 clock cycles
[1] TcC = TwCh + TwCl + TcC + TrC
[2] Increase delay by 10 nsec for each 50 pF increase in loading. 200 pF maximum for data lines and 100 pF for control lines
[3] Increase delay by 10 nsec for each 10 pF increase in loading. 100 pF maximum

OUTPUT LOAD CIRCUIT



Z80A-CTC A.C. Characteristics

$T_A = 0°C$ to $70°C$, $V_{CC} = +5\ V \pm 5\%$ unless otherwise noted

Number	Symbol	Parameter	Min	Max	Unit	Comments
1	TdC	Clock Cycle Time	400	[1]	ns	
2	TwCh	Clock Width (High)	170	2000	ns	
3	TwCL	Clock Width (Low)	170	2000	ns	
4	TfC	Clock Fall Time		30	ns	
5	TrC	Clock Rise Time		30	ns	
6	Th	All Hold Times	0		ns	
* 7	TsCS(C)	$\overline{CS}$ to Clock ↑ Setup Time	250		ns	
* 8	TsCE(C)	$\overline{CE}$ to Clock ↑ Setup Time	200		ns	
9	TsIO(C)	$\overline{IORQ}$ ↓ to Clock ↑ Setup Time	250		ns	
10	TsRD(C)	$\overline{RD}$ to Clock ↑ Setup Time	240		ns	
*11	TdC(DO)	Clock ↓ to Data Out Delay		240	ns	[2]
12	TdC(DOz)	Clock ↑ to Data Out Float Delay		230	ns	
13	TsDI(C)	Data In to Clock ↑ Setup Time	60		ns	
14	TsM1(C)	$\overline{M1}$ to Clock ↑ Setup Time (INTA or M1 Cycle)	210		ns	
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt immediately preceding M1) See Note A		300	ns	[3]
16	TdIO(DOT)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)		340	ns	[2]
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		190	ns	[3]
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)		220	ns	[3]
*19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		TcC+230	ns	Timer Mode
*20	TdCTK(INT)	CLR/TRG ↑ to INT ↓				
		TsCTR(C) Satisfied		TcC+230	ns	Counter Mode
		TsCTR(C) not Satisfied		2TcC+530	ns	
21	TcCTR	CLK Cycle Time	2TcC		ns	Counter Mode
22	TrCTR	CLK/TRG Rise Time		50	ns	
23	TfCTR	CLK/TRG Fall Time		50	ns	
24	TwCTRL	CLK/TRG Width (Low)	200		ns	
25	TwCTRh	CLK/TRG Width (High)	200		ns	
26	TsCTR(Cc)	CLK ↑ to Clock ↑ Setup Time for Immediate Count	300		ns	Counter Mode
27	TsCTR(Ct)	TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following Clock ↑	210		ns	Timer Mode
28	TdC(ZCTOr)	Clock ↑ to ZC/TO ↑ Delay	260		ns	
29	TdC(ZCTOf)	Clock ↓ to ZC/TO ↓ Delay	190		ns	

Notes A $2.5\ TcC > (N-2)\ TdIEI(IEOF)+TdM1(IEO)+TsIEI(IO)$

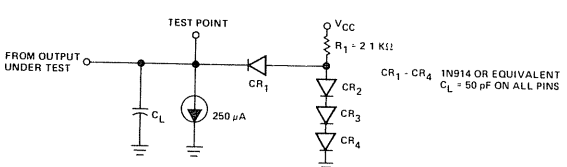
B $\overline{RESET}$ must be active for a minimum of 3 clock cycles

[1] $TcC = TwCh + TwCl + TcC + TrC$

[2] Increase delay by 10 nsec for each 50 pF increase in loading, 200 pF maximum for data lines and 100 pF for control lines

[3] Increase delay by 10 nsec for each 10 pF increase in loading, 100 pF maximum

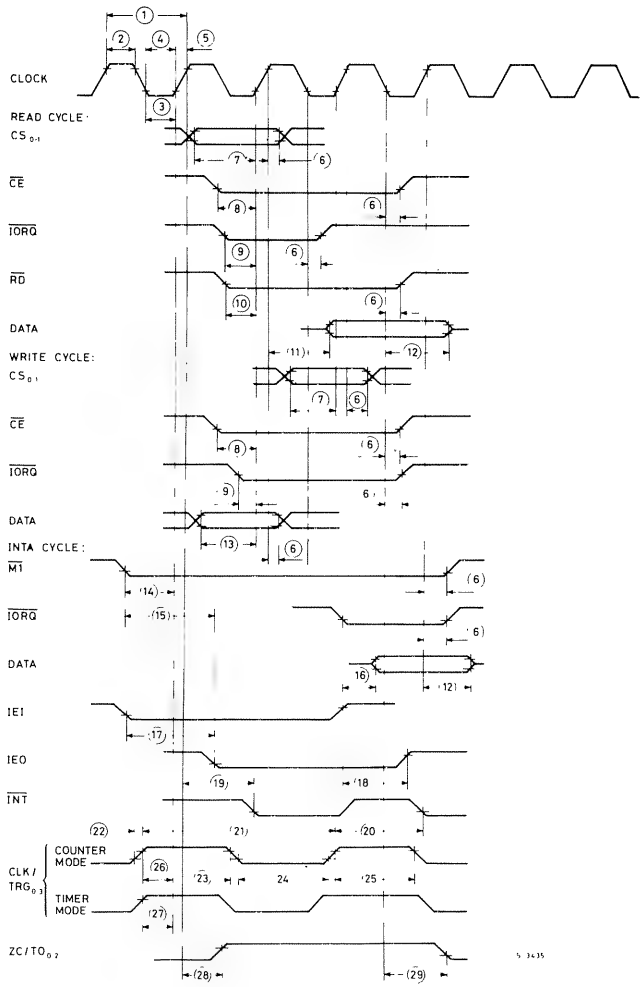
OUTPUT LOAD CIRCUIT



A.C. Timing Diagram

Timing measurements are made at the following voltages, unless otherwise specified:

	"1"	"0"
CLOCK	V_{CC}-0.6V	0.45V
OUTPUT	2 V	0.8V
INPUT	2 V	0.8V
FLOAT	ΔV	±0.5V



Absolute Maximum Ratings

Temperature Under Bias	0°C to 70°C
Storage Temperature	-65°C to +150°C
Voltage On Any Pin with Respect to Ground	-0.3V to +7V
Power Dissipation	0.8W

* Comment
Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Z80-CTC D.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= 5V ± 5% unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3	0.45	V	
V_{IHC}	Clock Input High Voltage [1]	V_{CC}-0.6	V_{CC}+0.3	V	
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{IH}	Input High Voltage	2	V_{CC}	V	
V_{OL}	Output Low Voltage		0.4	V	I_{OL} = 2 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = -250 μA
I_{CC}	Power Supply Current		120	mA	T_C = 400 nsec
I_{LI}	Input Leakage Current		10	μA	V_{IN} = 0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float		10	μA	V_{OUT} = 2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float		-10	μA	V_{OUT} = 0.4V
I_{OHD}	Darlington Drive Current	-1.5		mA	V_{OH} = 1.5V R_{EXT} = 390Ω

Z80A-CTC D.C. Characteristics

T_A= 0°C to 70°C, V_{CC}= 5V ± 5% unless otherwise specified

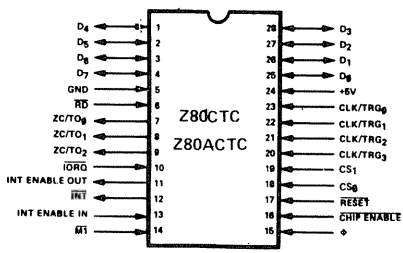
Symbol	Parameter	Min	Max	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3	0.45	V	
V_{IHC}	Clock Input High Voltage [1]	V_{CC}-0.6	V_{CC}+0.3	V	
V_{IL}	Input Low Voltage	-0.3	0.8	V	
V_{IH}	Input High Voltage	2	V_{CC}	V	
V_{OL}	Output Low Voltage		0.4	V	I_{OL} = 2 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = -250 μA
I_{CC}	Power Supply Current		120	mA	T_C = 250 nsec
I_{LI}	Input Leakage Current		10	μA	V_{IN} = 0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float		10	μA	V_{OUT} = 2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float		-10	μA	V_{OUT} = 0.4V
I_{OHD}	Darlington Drive Current	-1.5		mA	V_{OH} = 1.5V R_{EXT} = 390Ω

Capacitance

TA = 25° C, f = 1 MHz

Symbol	Parameter	Max.	Unit	Test Condition
C_Φ	Clock Capacitance	20	pF	Unmeasured Pins
C_{IN}	Input Capacitance	5	pF	Returned to Ground
C_{OUT}	Output Capacitance	10	pF	

PIN CONNECTIONS

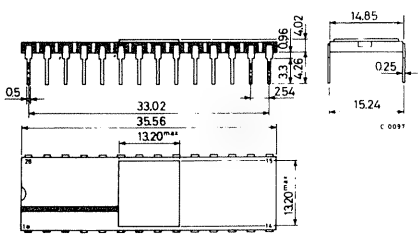


ORDERING NUMBERS:

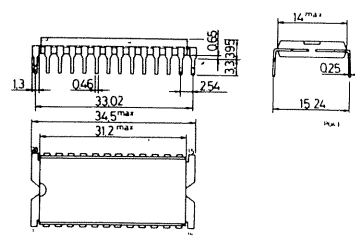
Z80CTC	D1	for dual in-line ceramic package (metal-seal)
Z80CTC	B1	for dual in-line plastic package
Z80ACTC	D1	for dual in-line ceramic package (metal-seal)
Z80ACTC	B1	for dual in-line plastic package
Z80CTC	F1	for dual in-line ceramic package (frit-seal)
Z80ACTC	F1	for dual in-line ceramic package (frit-seal)

MECHANICAL DATA (dimensions in mm)

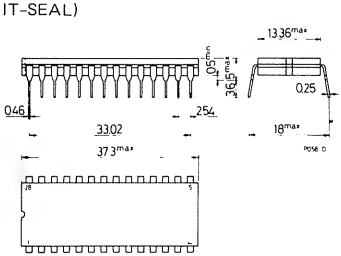
28-PIN CERAMIC DUAL IN-LINE SLAM PACKAGE



28-PIN PLASTIC DUAL IN-LINE PACKAGE



28-PIN CERAMIC DUAL IN-LINE PACKAGE (FRIT-SEAL)



Z 80-DMA
Z 80A-DMA

Product Specification

The Z-80 DMA (Direct Memory Access) circuit is a programmable single-channel device which provides all address, timing and control signals to effect the transfer of blocks of data between two ports within most microprocessor-based systems. These ports may be either system main memory or any system peripheral I/O device. The DMA can also search a block of data for a particular byte (bit maskable), with or without a simultaneous transfer.

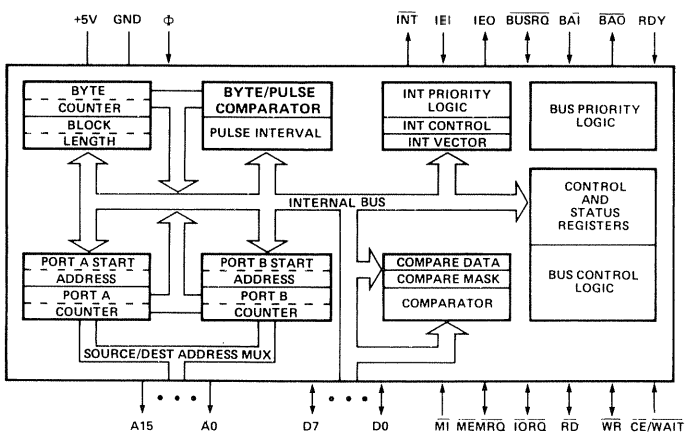
Structure

- N-channel Silicon Gate Depletion Load Technology
- 40 Pin DIP
- Single 5 volt supply
- Single phase 5 volt clock
- Single channel, two port

Features

- Three classes of operation:
  - –Transfer Only
  - –Search Only
  - –Search-Transfer
- Address and Block Length Registers fully buffered. Values for next operation may be loaded without disturbing current values.
- Dual addresses generated during a transfer (one for read port and one for write port).
- Programmable data transfers and searches, automatically incrementing or decrementing the port addresses from programmed starting addresses (they can also remain fixed).
- Three modes of operation:
  - –Byte-at-a-time: One byte transferred per request
  - –Burst: Continues as long as ports are ready
  - –Continuous: Locks out CPU until operation complete
- Timing may be programmed to match the speed of any port
- Interrupts on Match Found, End of Block, or Ready, may be programmed.
- An entire previous operation may be repeated automatically or on command. (Auto restart or Load)
- The DMA can signal when a specified number of bytes has been transferred, without halting transfer
- Multiple DMA's easily configured for rotating priority.
- The channel may be enabled, disabled or reset under software control
- Complete channel status upon program (CPU) request.
- Up to 1.25 megabyte/second Search.
- Daisy-chain priority interrupt and bus acknowledge included to provide automatic interrupt vectoring and bus request control, without need for additional external logic
- TTL compatible inputs and outputs
- The CPU can read current Port counters, Byte counter, or Status Register. A mask byte can be set which defines which registers can be accessed during read operations.

Fig. 7 - DMA INTERNAL BLOCK DIAGRAM



DMA Architecture

A block diagram of the Z80 DMA is shown in Figure 7. The internal structure consists of the following circuitry:

- *Bus Interface*: provides driver and receiver circuitry to interface to the Z80-CPU Bus
- *Control Logic and Registers*: set the class, mode and other basic control parameters of the DMA
- *Address, Byte Count and Pulse Circuitry*: generates the proper port addresses for the read and write operations, with provisions for incrementing or decrementing the address. When zero bytes remain to be handled, the byte count circuitry sets a flag in the status register. Pulse circuitry generates a pulse each time the byte counter lower 8-bits equal the pulse register.
- *Timing Circuitry*: allows the user to completely specify the read/write timing for each port.
- *Match Circuitry*: holds the match byte and a mask byte which allows for the comparison of only certain bits within the byte. If a match is encountered during a Search or Transfer, this circuitry sets a flag in the status register.
- *$\overline{INT}$ and $\overline{BUSRQ}$ Circuitry*: includes a control register which specifies the conditions under which the DMA can generate an interrupt; priority encoding logic to select between the generation of an $\overline{INT}$ or $\overline{BUSRQ}$ output under these conditions; and an interrupt vector register for automatic vectoring to the interrupt service routine
- *Status Register*: holds current status of DMA.

Register Description

The following DMA-internal registers are available to the programmer:

Control Registers: Write only; 8 bits. Hold DMA control information: such as, when to initiate an interrupt or pulse, what mode or class of operation to perform, etc.

Timing Registers: Write only; 8 bits. Hold read/write timing parameters for the two ports.

Interrupt Vector Register: Read/write; 8 bits. Holds the 8-bit vector that the DMA will put onto the data bus after receiving an $\overline{IORQ}$ during an interrupt acknowledge sequence if it is the highest priority device requesting an interrupt. (This register is readable only during interrupt acknowledge cycles.)

Block Length Register: Write only; 16 bits. Contains total block length of data to be searched and/or transferred.

Byte Counter: Read only; 16 bits. Counts number of bytes transferred (or searched). On a Load or Continue the Byte Counter is reset to zero. Thereafter, each byte transfer operation increments it until it matches the contents of the Block Length Register, at which time End of Block is set in the status register and operation is suspended if programmed. Also if so programmed the DMA will generate an interrupt.

Match Register: Write only; 8 bits. Holds the byte for which a match is being sought in Search operations.

Mask Register: Write only; 8 bits. Holds the 8-bit mask to determine which bits in the match register are to be examined for a match.

Starting Address Registers (Port A and Port B): Write only; 16 bits each. Hold the starting addresses (upper and lower 8 bits) for the two ports involved in Transfer operations. In Search Only operations, only one port address would have to be specified. Only memory starting addresses require both upper and lower 8 bits; I/O ports are generally addressed with only the lower 8 bits, and in this case the address contained in the register is a generally fixed address.

Address Counters (Port A and Port B): Read only; 16 bits each. These counters are loaded with the contents of the corresponding Starting Address Registers whenever Searches or Transfers are initiated with a Load or Continue. They are incremented, decremented or remain fixed, as programmed.

Pulse Control Register: Write only; 8 bits. The content of this register is continuously compared with the lower eight bits of the byte counter. When they become equal, the $\overline{INT}$ output is activated. Since this occurs while $\overline{BUSRQ}$ and $\overline{BUSAK}$ are both active, the CPU does not interpret this as an interrupt request. Instead, the signal is used to communicate with a peripheral I/O device. When the Pulse Control Register contains a value n, the first pulse is generated after n + 1 bytes of search or transfer. The next and all subsequent pulses occur at 256-byte intervals.

Status Register: Read only; 8 bits. Match, End of Block, Ready Active, Interrupt Pending, and DMA Cycle Occurred bits indicate these functions when set.

Modes of Operation

The DMA may be programmed for one of three modes of operation. (See Command Register 2B.)

- *Byte at a time*: control is returned to the CPU after each one-byte cycle.
- *Burst*: operation continues as long as the DMA's RDY input is active, indicating that the relevant port is ready. Control returns to the CPU when RDY is inactive or at end of block or a match if so programmed.
- *Continuous*: the entire Search and/or Transfer of a block of data is completed before control is returned to CPU

Classes of Operation

The DMA has three classes of operation: Transfer only, Search Only and a combined Search-Transfer. (See Command Register 1A.)

During a Transfer, data is first read from one port and then written to the other port, byte by byte. (The DMA's two ports are termed Port A and Port B.) The ports may be programmed to be either system main memory or peripheral I/O devices. Thus, a block of data might be written from a peripheral to another; or it might be written from one area in main memory to another; or from a peripheral to main memory.

During a Search, data is read only, and compared byte by byte against two DMA-internal registers, one of which contains a match byte and the other an optional mask byte which allows only certain bits to be compared. If any byte of searched data matches, a DMA-internal status bit is set; if programmed to do so, the DMA will then suspend operation and/or generate an interrupt.

The third class of operation is a combined Search-Transfer. In such an operation a block of data is transferred as described above until a match is found; then, as in a Search Only operation, the transfer may be suspended and/or an interrupt generated

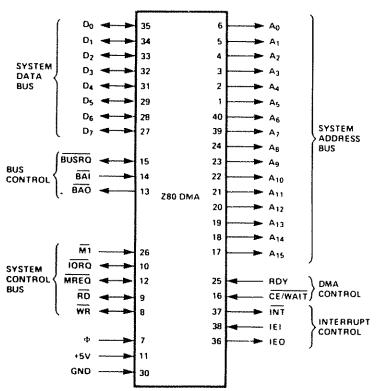
Addressing

The DMA's addressing of ports is either fixed or sequential, incrementing or decrementing from a starting address. The length of the operation (number of bytes) is specified by the programmed contents of a block length register. The DMA can address block lengths of up to 64K bytes. During a transfer two separate port addresses are generated, one during the Read cycle and one during the Write cycle.

Operating Sequence

Once the DMA has been programmed it may be "Enabled" (Command Register 2A or 2D). In the enabled condition when Ready goes active the DMA will request the bus by bringing $\overline{BUSRQ}$ low. The CPU will acknowledge this with a $\overline{BUSACK}$ which will normally be attached to $\overline{BAI}$. When the DMA receives $\overline{BAI}$ it will start its programmed operation releasing $\overline{BUSRQ}$ to a "high" state when it is through.

Z-80 DMA Pin Description



A_0-A_{15} System Address Bus. All sixteen of these pins are used by the DMA to address system main memory or an I/O port (output)

D_0-D_7 System Data Bus. Commands from the CPU, DMA status and data from memory or peripherals are transferred on these tristate pins (input/output)

+5V Power

GND Ground

Φ System clock (input)

$\overline{M1}$ Machine cycle One signal from CPU (input)

$\overline{IORQ}$ Input/Output Request to and from the System Bus (input/output)

$\overline{MREQ}$ Memory REQuest to the System Bus (input/output)

$\overline{RD}$ ReaD to and from the System Bus (input/output)

$\overline{WR}$ WRite to and from the System Bus (input/output)

$\overline{CE/WAIT}$ Chip Enable; may also be programmed to be $\overline{WAIT}$ during time when $\overline{BAI}$ is low (input)

$\overline{BUSRQ}$ BUS ReQuest. Requests control of the CPU Address Bus, Data Bus and Status/Control Bus (input/output, open drain)

$\overline{BAI}$ Bus Acknowledge In. Signals that the system buses have been released for DMA control (input)

$\overline{BAO}$ Bus Acknowledge Out. $\overline{BAI}$ and $\overline{BAO}$ form a daisy-chain connection for system-wide priority bus control (output)

$\overline{INT}$ INTerrupt request (output, open drain)

IEI Interrupt Enable In (input)

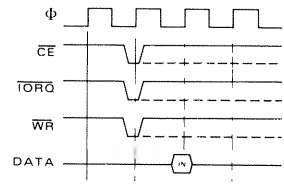
IEO Interrupt Enable Out. IEI and IEO form a daisy-chain connection for system-wide priority interrupt control (output)

RDY ReaDY is monitored by the DMA to determine when a peripheral device associated with a DMA port is ready for a read or write operation (input, programmable as active high or low)

DMA Timing Waveforms

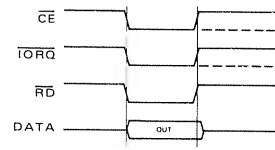
DMA Command Write Cycle

Illustrated here is the timing associated with a command byte or control byte being written to the DMA which is to be loaded into internal registers. Z80 Output instructions satisfy this timing.



DMA Register Read Cycle

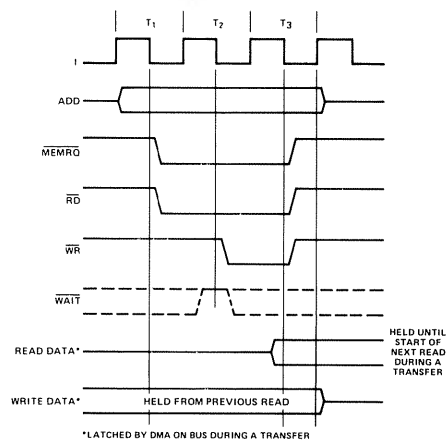
This timing is used when a read operation is performed on the DMA to access the contents of the Status Register, Address Counter or other readable registers Z80 Input instructions satisfy this timing.



STD Memory Timing

This timing is exactly the same as used by the Z80-CPU to access system main memory, either in a Read or Write operation. The DMA will default to this timing after a power-on reset, or when a Reset or Reset Timing command is written to it, and unless otherwise programmed, will use this timing during all Transfer or Search operations involving system main memory. During the memory Read portion of a transfer cycle, data is latched in the DMA on the negative edge of Φ during T_3 and held into the following Write cycle During the memory Write portion of a transfer cycle, data is held from the previous Read cycle and released at the end of the present cycle.

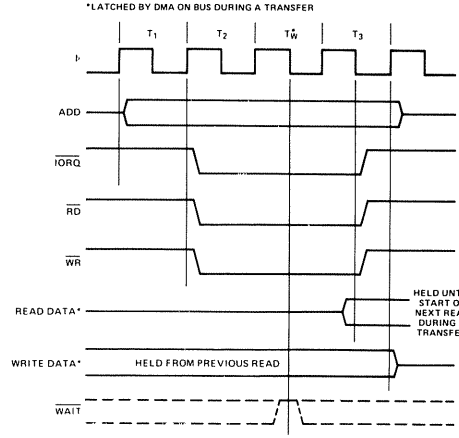
NOTE: The DMA is normally programmed for a 3 T-cycle duration in memory transactions. But $\overline{WAIT}$ is sampled during the negative transition of T_2, and if it is low, T_2 will be extended another T-cycle, during which $\overline{WAIT}$ will again be sampled. The duration of a memory transaction cycle may thus be indefinitely extended.



STD Peripheral Timing

This timing is identical to the Z80-CPU's Read/Write timing to I/O peripheral devices. The DMA will default to this timing after a power-on reset, or when a Reset or Reset Timing command is written to it, and unless otherwise programmed, will use this timing during all Transfer or Search operations involving I/O peripherals. During the I/O Read of a transfer cycle, data is latched on the negative edge of Φ during T_3 and is then held into the Write cycle. During an I/O Write, data is held from the previous Read cycle until the end of the Write cycle.

NOTE. If $\overline{WAIT}$ is low during the negative transition of T_W*, then T_W* will be extended another T-cycle and $\overline{WAIT}$ will again be sampled. The duration of a peripheral transaction cycle may thus be indefinitely extended

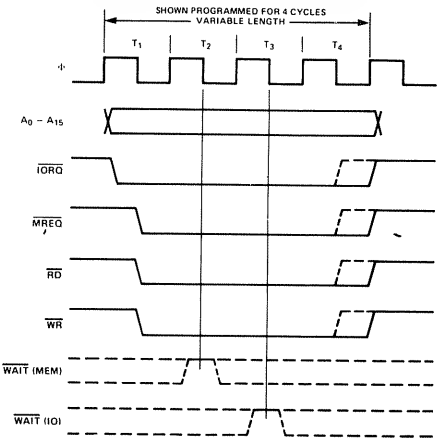


DMA Timing Waveforms (continued)

Variable Cycle

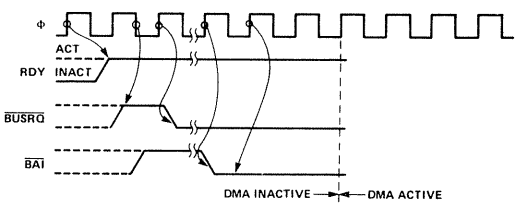
The Variable feature of the DMA allows the user to program the DMA's memory or peripheral transaction timing to values different than given above in the standard default diagrams. This permits the designer to tailor his timing to the particular requirements of his system components, and maximizes the data transfer rate while eliminating external signal conditioning logic. Cycle length can be two to four T-cycles (more if $\overline{WAIT}$ is used). Signal timing can be varied as shown. During a transfer, data will be latched by the DMA on the clock edge causing the rising edge of $\overline{RD}$ and will be held on the data lines until the end of the following Write cycle.

(See Timing Control Byte, page 9).



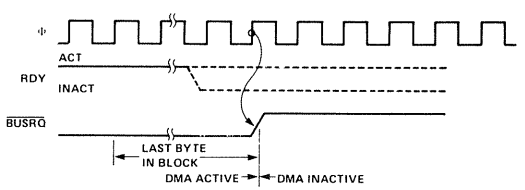
DMA Bus Request and Acceptance for Byte-at-a-Time, Burst, and Continuous Mode

Ready is sampled on every rising edge of Φ. When it is found to be active, the following rising edge of Φ generates $\overline{BUSRQ}$. After receiving $\overline{BUSRQ}$ the CPU will grant a $\overline{BUSAK}$ which will be connected to $\overline{BAI}$ either directly or through the Bus Acknowledge Daisy Chain. When a low is detected on $\overline{BAI}$ for two consecutive edges of Φ, the next rising edge of Φ will start an active DMA cycle.



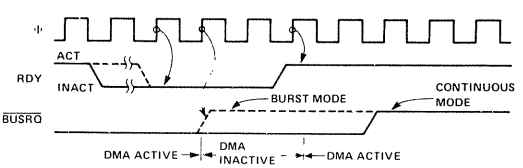
DMA Bus Release at End of Block for Burst or Continuous Mode

Timing for End of Block and DMA not programmed for Auto-restart.



DMA Bus Release with 'Ready' for Burst and Continuous Mode

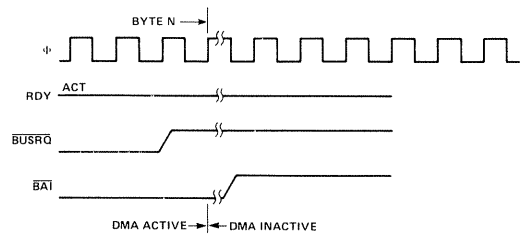
The DMA will relinquish the bus after RDY has gone inactive (Burst mode) or after an End of Block or a Match is found (Continuous mode) With RDY inactive, the DMA in Continuous mode is inactive but maintains control of the bus ($\overline{BUSRQ}$ low) until the cycle is resumed when RDY goes active.



Reading from the DMA Internal Registers

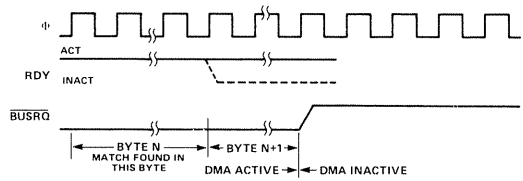
DMA Bus Release for Byte-at-a-Time Mode

In the Byte mode the DMA will release $\overline{BUSRQ}$ on the rising edge of Φ prior to the end of each Read cycle in Search Only or each Write cycle in a Transfer, regardless of the state of RDY. The next bus request will come after both $\overline{BUSRQ}$ and $\overline{BAI}$ have returned high.



DMA Bus Release with Match for Burst or Continuous Modes

When a Match is found and the DMA is programmed to stop on Compare, the DMA performs an operation on the next byte and then releases bus.



Reading the DMA Internal Registers

The CPU can read seven internal DMA registers, always in the following order: Status, lower byte of the Block Length register, upper byte of the Block Length register, lower byte of the Port A Address, upper byte of the Port A Address, lower byte of the Port B Address and the upper byte of the Port B Address.

The Read Mask register must be programmed to either include or exclude any of these seven registers by programming a 1 (include) or 0 (exclude) in the appropriate positions of the Read Mask register. After a Reset or Load, the read sequence must be initiated through an Initiate Read Sequence command (Command Byte 2D). The sequence of reading all registers that are not excluded by the Read Mask register must be completed before a new Initiate Read Sequence or RD Status command.

Programming the DMA

Previous sections of this specification have indicated the various functions and modes of the DMA. The diagrams and charts below show how the DMA is programmed to select among these functions and modes and to adapt itself to the requirements of the user system.

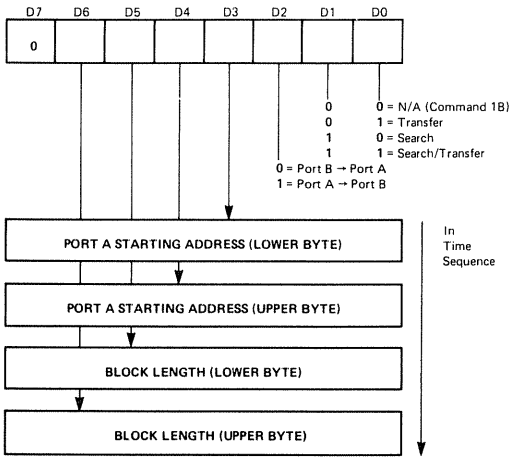
The Z80-DMA chip may be in an "enable" state, in which it can gain control of the system buses and direct the transfer of data between its ports, or in a "disable" state, when it cannot gain control of the bus Program commands can be written to it in either state, but writing a command to it automatically puts it in the disable state, which is maintained until an enable command is issued to the DMA. The CPU must program it in advance of any data search or transfer by addressing it as an I/O port and sending it a sequence of command bytes via the system data bus using Output instructions. When the DMA is powered up or reset by any means, the DMA is automatically placed into a disable state, in which it can initiate neither bus requests nor data transfers nor interrupts.

The command bytes contain information to be loaded into the DMA's control and other registers and/or information to alter the state of the chip, such as an Enable Interrupt command. The command structure is designed so that certain bits in some commands can be set to alert the DMA to expect the next byte written to it to be for a particular internal register.

The following diagrams and charts give the function of each bit in the six different command bytes. Two of these are defined as being from Group 1, and are termed command bytes 1A and 1B These Group 1 commands contain the most basic DMA set-up information The other four are categorized as Group 2, and are termed commands 2A, 2B, 2C and 2D. Group 2 words specify more detailed set-up information

Programming the DMA (continued)

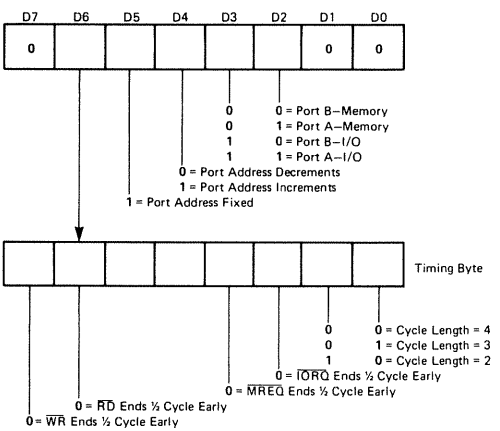
Command Register 1A



A "1" in positions D_3 through D_6 means that the indicated byte will follow. Note that the sequence of bytes is absolutely rigid.

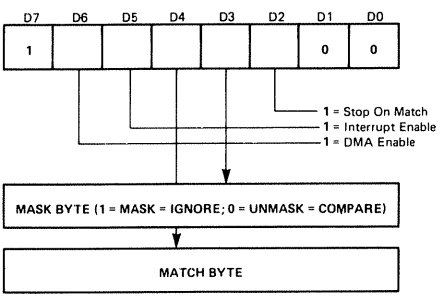
The DMA always transfers or searches one byte more than the number written into the Block Length registers. A "0" in the block length register results in the transfer or search of 2^{16} + 1 bytes. The shortest programmable block length is therefore two bytes long, programmed by writing a 1 into the Block Length register.

Command Register 1B

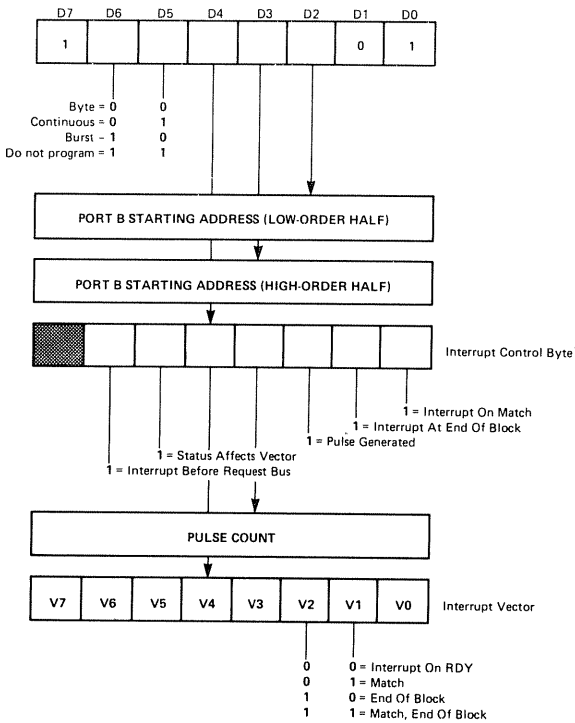


For transfers, this byte is normally written twice, once for Port A and again for Port B.

Command Register 2A



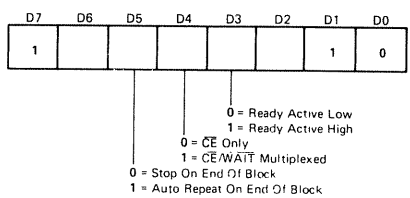
Command Register 2B



[1] If "Interrupt Before Requesting Bus" is selected (by a 1 in bit 6 of the Interrupt Control byte), the Z-80 DMA does not request the bus until the following set of instructions has been received by the Z-80 DMA:

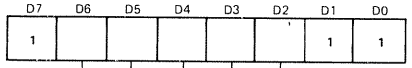
- Enable after RETI command (B7 in Command byte 2D)
- Enable DMA command (87 in Command byte 2D)
- A RETI instruction that resets the IUS (Interrupt Under Service latch) in the Z-80 DMA

Command Register 2C

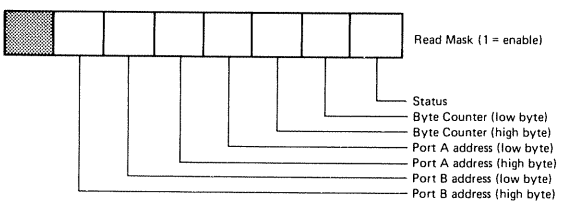


Programming the DMA (continued)

Command Register 2D



HEX	D6	D5	D4	D3	
C3	1	0	0	0	0 = **Reset** Interrupt circuitry disable interrupt and bus request logic, unforce internal ready condition, disable "MUXCE" and stop auto repeat
C7	1	0	0	0	1 = **Reset Port A Timing** to standard Z-80 CPU timing
CB	1	0	0	1	0 = **Reset Port B Timing** to standard Z 80 CPU timing
CF	1	0	0	1	1 = **Load** starting address for both ports, clear byte counter *
D3	1	0	1	0	0 = Addresses **continue** from present locations, clear byte counter
AB	0	1	0	1	0 = **Enable interrupts**
AF	0	1	0	1	1 = **Disable interrupts**
A3	0	1	0	0	0 = **Reset and disable interrupt** circuits (like RETI) and unforce the internal ready condition
87	0	0	0	0	1 = **Enable DMA**] Both affect all operations except interrupts, but do not
83	0	0	0	0	0 = **Disable DMA**] reset any functions
A7	0	1	0	0	1 = **Initiate read** sequence to the first register designated as readable by the Read Mask register
BF	0	1	1	1	1 = Set **read status** so next read is from status register
B3	0	1	1	0	0 = **Force** an internal **ready** condition independent of the RDY input Used for memory-to-memory operations where no RDY signal is needed This command does not function in the "byte-at a time" mode
8B	0	0	0	1	0 = **Clear** Match and End of Block **status** bits
B7	0	1	1	0	1 = **Enable after RETI** so DMA will request bus only after receiving a RETI Must be followed by an Enable DMA command
BB	0	1	1	1	0 = **Read mask** is the following byte



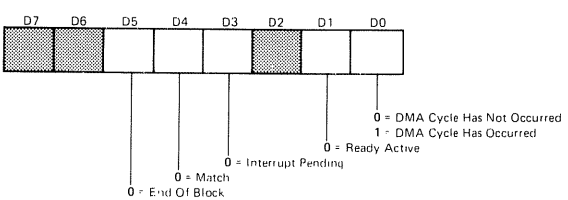
* Loading Port Addresses The "Load" command (CF in Command Register 2D) loads a fixed address only into a port selected as the source, not into a port selected as the destination. Therefore, the destination address must be loaded by temporarily mislabeling the destination as the source

The following example is a set-up procedure for a transfer from Port A to Port B:

1. Command byte 1A with B as source port
2. Command byte 2D with CF = load
3. Command byte 1A with A as source port
4. Command byte 2D with CF = load
5. Command byte 2D with 87 = Enable DMA

This manipulation is required only when the destination has a fixed address.

Status Register



Programming the DMA (continued)

The Sample DMA Program shows how the DMA may be programmed to transfer data from memory (Port A) to a peripheral device (Port B). In this example, the Port A memory starting address is 1050_H and the Port B peripheral fixed address is 05_H. Note that the data flow is 1001_H bytes—one more than specified by the block length. The table of DMA commands may be stored in consecutive memory locations and transferred to the DMA with an output instruction such as OTIR.

Sample DMA Program

	D7	D6	D5	D4	D3	D2	D1	D0	HEX
1) Command Register 1A sets DMA to receive block length, Port A starting address and temporarily sets Port B as source	0 Group One	1 Block Length Upper Follows	1 Block Length Lower Follows	1 Port A Upper Addr Follows	1 Port A Lower Addr Follows	0 B →A Temporary For Loading B Address	0 1 Command Byte 1A Transfer, No Search		79
2) Port A address (lower) 3) Port A address (upper) 4) Block length (lower) 5) Block length (upper)	0 0 0 0	1 0 0 0	0 0 0 0	1 1 0 1	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	50 10 00 10
6) Command Register 1B defines Port A as memory with incrementing address	0 Group One	0 No Timing Follows	0 Address Changes	1 Address Increments	0 Port Is Memory	1 This Is Port A	0 0 Byte 1B		14
7) Command Register 1B defines Port B as peripheral with fixed address	0 Group One	0 No Timing Follows	1 Fixed Address	0 Not Used	1 Port Is I/O	0 This Is Port B	0 0 Byte 1B		28
8) Command Register 2B sets mode to Burst, sets DMA to expect Port B address	1 Group Two	1 0 Burst Mode		0 No Interrupt Control Byte Follows	0 No Upper Address	1 Port B Lower Addr Follows	0 1 Byte 2B		C5
9) Port B address (lower)	0	0	0	0	0	1	0	1	05
10) Command Register 2C sets Ready active High	1 Group Two	0 Not Used	0 No Auto Restart	0 No Wait States	1 RDY Active HIGH	0 Not Used	1 0 Byte 2C		8A
11) Command Register 2D loads Port B address and resets block counter	1 Group Two	1	0	0 Load	1	1	1 1 Byte 2D		CF
12) Command Register 1A sets Port A as source *	0 Group One	0	0	0 No Addr Or Block Length Bytes	0	1 A →B	0 1 Byte 1A, Transfer No Search		03
13) Command Register 2D loads Port A address and resets block counter *	1 Group Two	1	0	0 Load	1	1	1 1 Byte 2D		CF
14) Command byte 2D enables DMA to start operation	1 Group Two	0	0	0 Enable DMA		1	1 1 Byte 2D		87

NOTE: The actual number of bytes transferred is one more than specified by the block length
* These commands are necessary only in the case of a fixed destination address

Absolute Maximum Ratings

Temperature Under Bias	0°C to +70°C
Storage Temperature	−65°C to +150°C
Voltage On Any Pin with Respect to Ground	−0.3V to +7V
Power Dissipation	1.5W

*Comment

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. Characteristics

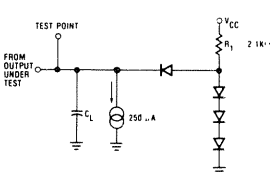
T_A = 0°C to 70°C, V_{CC} = 5V ±5% unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	−0 3	0 45	V	
V_{IHC}	Clock Input High Voltage	V_{CC}−6	5 5	V	
V_{IL}	Input Low Voltage	−0 3	0 8	V	
V_{IH}	Input High Voltage	2 0	5 5	V	
V_{OL}	Output Low Voltage		0 4	V	I_{OL} = 3 2 mA for $\overline{BUSRQ}$ I_{OL} = 2 0 mA for all others
V_{OH}	Output High Voltage	2 4		V	I_{OH} = − 250 μA
I_{CC}	Power Supply Current Z-80 DMA Z-80A DMA		150 200	mA mA	t_C = 400 ns t_C = 250 ns
I_{LI}	Input Leakage Current		10	μA	V_{IN} = 0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float		10	μA	V_{OUT} = 2 4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float		−10	μA	V_{OUT} = 0 4V
I_{LD}	Data Bus Leakage Current in Input Mode		±10	μA	$0 \leqslant V_{IN} \leqslant V_{CC}$

Capacitance

T_A = 25°C, f = 1 MHz

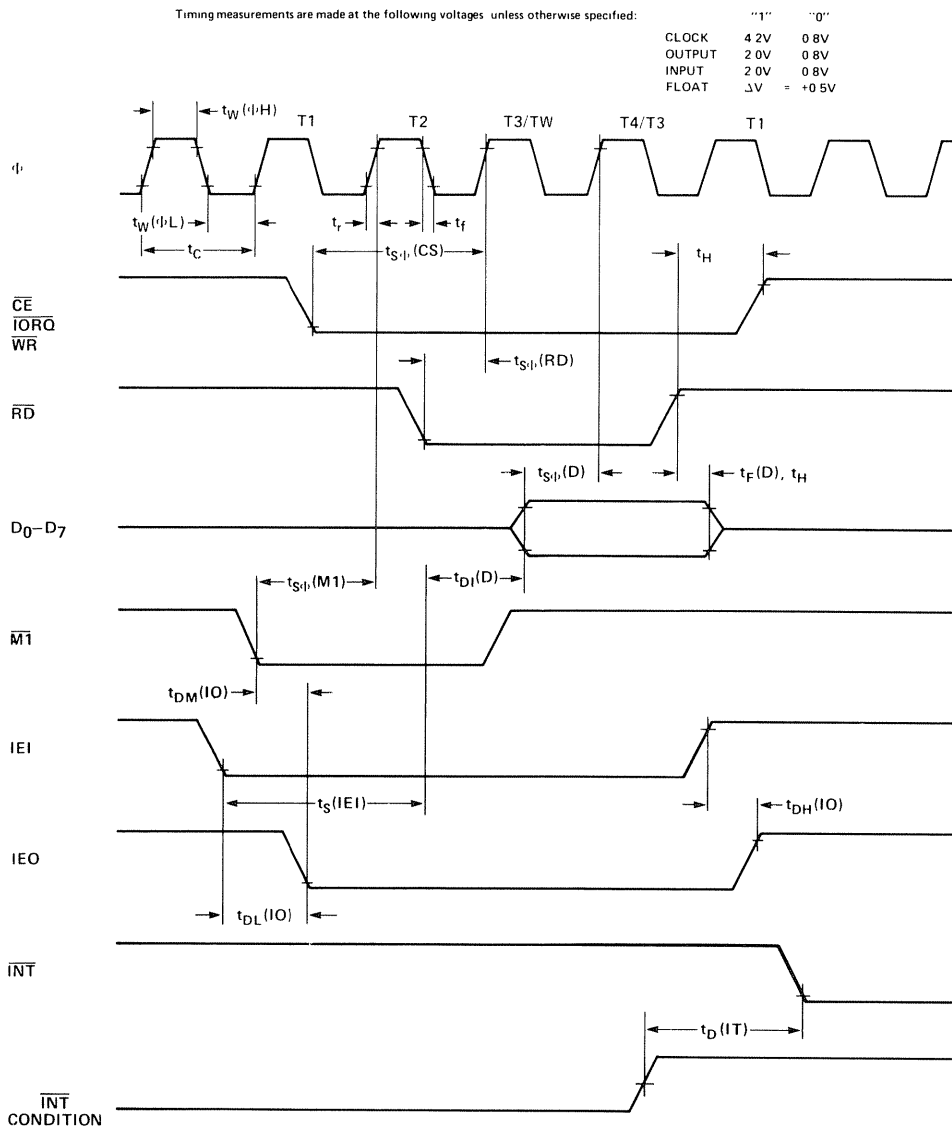
Symbol	Parameter	Max.	Unit	Test Condition
C_Φ	Clock Capacitance	35	pF	Unmeasured Pins
C_{IN}	Input Capacitance	5	pF	Returned to Ground
C_{OUT}	Output Capacitance	10	pF	



C_L = 50 pF. Increase delay by 10 ns for each 50 pF increase in C_L, up to 200 pF maximum.

A.C. Timing Diagrams

Z80 and Z80A as a Peripheral Device (Inactive State)



NOTE: This diagram does not show an actual timing sequence. Refer to this diagram only for the detailed timing relationships of individual edges. Use the illustrations in the "DMA Timing Waveforms" section as an explanation of the various timing sequences.

A.C. Characteristics

Z80-DMA as a Peripheral Device (Inactive State)
$T_A = 0°C$ to $70°C$, Vcc = +5V±5%, Unless Otherwise Noted

SIGNAL	SYMBOL	PARAMETER	Z 80 DMA		Z 80A DMA*		UNIT
			MIN	MAX	MIN	MAX	
Φ	t_c $t_{w(\Phi H)}$ $t_{w(\Phi L)}$ t_r, t_f	Clock Period Clock Pulse Width, Clock High Clock Pulse Width, Clock Low Clock Rise and Fall Times	400 170 170	[1] 2000 2000 30	250 105 105	[1] 2000 2000 30	nsec nsec nsec nsec
	t_H	Any Hold Time for Specified Setup Time	0		0		nsec
CE, WR IORQ	$t_{S(\Phi)CS}$	Control Signal Setup Time to Rising Edge of Φ during Write Cycle (IORQ, WR, CE)	280		145		nsec
D_0 /	$t_{DR(D)}$ $t_{S(\Phi)D}$ Cycle $t_{D(D)}$ Cycle $t_{F(D)}$	Data Output Delay from Falling Edge of RD Data Setup Time to Rising Edge of Φ during Write or M1 Data Output Delay from Falling Edge of IORQ during INTA Delay to Floating Bus (Output Buffer Disable Time)	50	500 340 160	50	380 160 110	nsec nsec nsec nsec
IEI	$t_{S(IEI)}$	IEI Setup Time to Falling Edge of IORQ during INTA Cycle	140		140		nsec
IEO	$t_{DH(IEI)}$ $t_{DL(IO)}$ $t_{DM(IO)}$	IEO Delay Time from Rising Edge of IEI IEO Delay Time from Falling Edge of IEI IEO Delay from Falling Edge of M1 (Interrupt Occurring Just Prior to M1) See Note A		210 190 300		160 130 190	nsec nsec nsec
M1	$t_{S(\Phi M1)}$	M1 Setup Time to Rising Edge of Φ during INTA or M1 Cycle See Note B	210		90		nsec
RD	$t_{S(\Phi RD)}$	RD Setup Time to Rising Edge of Φ during M1 Cycle	240		115		nsec
INT	$t_{D(IT)}$	INT Delay Time from Condition Causing INT INT generated only when DMA is inactive		500		500	nsec
BAO	$t_{DH(BO)}$ $t_{DL(BO)}$	BAO Delay from Rising Edge of BAI BAO Delay from Falling Edge of BAI		200 200		150 150	nsec nsec

* Z-80A DMA Timing Specifications are preliminary

Z80-DMA as a Bus Controller (Active State)
$T_A = 0°C$ to $70°C$, Vcc = +5V±5%, Unless Otherwise Noted

SIGNAL	SYMBOL	PARAMETER	Z-80 DMA		Z-80A DMA		UNIT
			MIN	MAX	MIN	MAX	
Φ	t_C $t_{W(\Phi H)}$ $t_{W(\Phi L)}$ t_r, t_f	Clock Period Clock Pulse Width, Clock High Clock Pulse Width, Clock Low Clock Rise and Fall Time	400 180 180	 2000 2000 30	250 110 110	 2000 2000 30	nsec nsec nsec nsec
A_0-A_{15}	$t_{D(AD)}$ $t_{F(AD)}$ t_{acm} t_{aci} t_{ca} t_{caf}	Address Output Delay Delay to Float Address Stable Prior to MREQ (Memory Cycle) Address Stable Prior to IORQ, RD or WR (I/O Cycle) Address Stable from RD or WR Address Stable from RD or WR during Float	 $t_W(\Phi H) + t_f - 75$ $t_C - 80$ $t_W(\Phi L) + t_r - 40$ $t_W(\Phi L) + t_r - 60$	145 110 	 $t_W(\Phi H) + t_f - 65$ $t_C - 70$ $t_W(\Phi L) + t_r - 50$ $t_W(\Phi L) + t_r - 45$	110 90 	nsec nsec nsec nsec nsec nsec
D_0-D_7	$t_{D(D)}$ $t_{F(D)}$ $t_{S(H)D}$ $t_{S(F)D}$ t_{dcm} t_{dci} t_{cdf}	Data Output Delay Delay to Float during Write Cycle Data Setup Time to Rising Edge of Clock during Read when Falling Edge Ends RD Data Setup Time to Falling Edge of Clock during Read when Rising Edge Ends RD Data Stable Prior to WR (Memory Cycle) Data Stable Prior to WR (I/O Cycle) Data Stable from WR	 50 60 $t_C - 210$ $t_W(\Phi L) + t_r - 210$ $t_W(\Phi L) + t_r - 80$	 230 90 	 $t_C - 170$ $t_W(\Phi L) + t_r - 170$ $t_W(\Phi L) + t_r - 70$	 150 90 	nsec nsec nsec nsec nsec nsec nsec
	t_H	Any Hold Time where Setup Time is Specified	0			0	nsec
MREQ	$t_{DL\Phi(MR)}$ $t_{DH\Phi(MR)}$ $t_{DH\Phi(MR)}$ $t_{DH\Phi(MR)}$ $t_{W(MRL)}$ $t_{W(MRH)}$	MREQ Delay from Rising Edge of Clock MREQ Low MREQ Delay from Falling Edge of Clock MREQ Low MREQ Delay from Rising Edge of Clock MREQ High MREQ Delay from Falling Edge of Clock MREQ High Pulse Width MREQ Low Pulse Width MREQ High	 $t_C - 40$ $t_W(\Phi H) + t_f - 30$	 100 100 100 100 	 $t_C - 30$ $t_W(\Phi H) + t_f - 20$	 85 85 85 85 	nsec nsec nsec nsec nsec nsec
IORQ	$t_{DL\Phi(IR)}$ $t_{DL\Phi(IR)}$ $t_{DH\Phi(IR)}$ $t_{DH\Phi(IR)}$	IORQ Delay from Rising Edge of Clock IORQ Low IORQ Delay from Falling Edge of Clock IORQ Low IORQ Delay from Rising Edge of Clock IORQ High IORQ Delay from Falling Edge of Clock IORQ Low		90 110 100 110		75 85 85 85	nsec nsec nsec nsec
RD	$t_{DL\Phi(RD)}$ $t_{DL\Phi(RD)}$ $t_{DH\Phi(RD)}$ $t_{DH\Phi(RD)}$	RD Delay from Rising Edge of Clock RD Low RD Delay from Falling Edge of Clock RD Low RD Delay from Rising Edge of Clock RD High RD Delay from Falling Edge of Clock RD High		100 130 100 110		85 95 85 85	nsec nsec nsec nsec
WR	$t_{DL\Phi(WR)}$ $t_{DL\Phi(WR)}$ $t_{DH\Phi(WR)}$ $t_{DH\Phi(WR)}$ $t_{W(WRL)}$	WR Delay from Rising Edge of Clock WR Low WR Delay from Falling Edge of Clock WR Low WR Delay from Rising Edge of Clock WR High WR Delay from Falling Edge of Clock WR High Pulse Width WR Low	 $t_C - 40$	80 90 100 100 	 $t_C - 30$	65 80 80 80 	nsec nsec nsec nsec nsec
WAIT	$t_{S(WT)}$	WAIT Setup Time to Falling Edge of Clock	70		70		nsec
BUSRQ	$t_{D(BQ)}$	BUSRQ Delay Time from Rising Edge of Clock		100		100	nsec
	$t_{F(C)}$	Delay to Float (MREQ, IORQ, RD and WR)		100		80	nsec

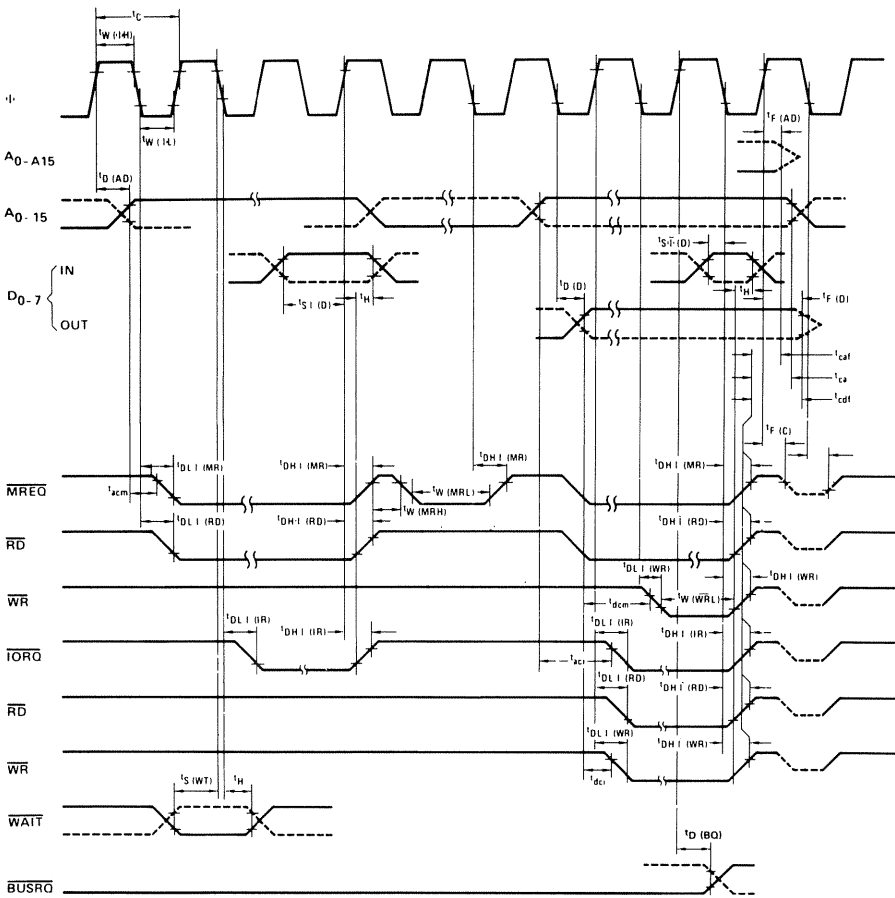
NOTES: ■ Data must be enabled onto the DMA data bus when RD is active. ■ Z-80A DMA timing specifications are preliminary ■ All equations imply standard Z-80 CPU and Z-80A CPU

A.C. Timing Diagrams

Z80 and Z80A as a Bus Controller (Active State)

Timing measurements are made at the following voltages, unless otherwise specified:

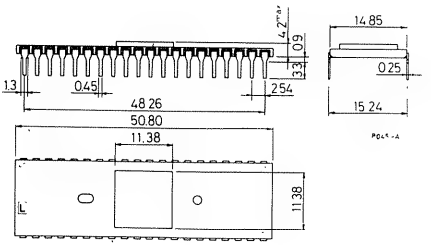
	"1"	"0"
CLOCK	4.2V	0.8V
OUTPUT	2.0V	0.8V
INPUT	2.0V	0.8V
FLOAT	ΔV =	+0.5V



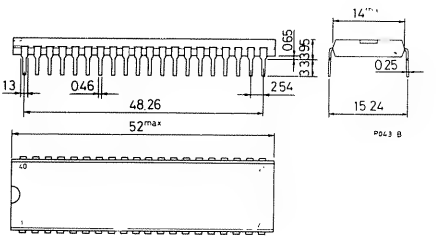
This diagram does not show an actual timing sequence. Refer to this diagram only for the detailing timing relationships of individual edges.

MECHANICAL DATA (dimensions in mm)

40-PIN CERAMIC DUAL IN-LINE SLAM PACKAGE



40-PIN PLASTIC DUAL IN-LINE PACKAGE



ORDERING NUMBERS:

Z80-DMA D1 for dual in-line ceramic slam package
Z80-DMA B1 for dual in-line plastic package
Z80A-DMA D1 for dual in-line ceramic slam package
Z80A-DMA B1 for dual in-line plastic package

Z 80-SIO
Z 80A-SIO

Product Specification

General Description

The Z80-SIO (Serial Input/Output) is a dual-channel multi-function peripheral component designed to satisfy a wide variety of serial data communications requirements in microcomputer systems. Its basic function is a serial-to-parallel, parallel-to-serial converter/controller, but—within that role—it is configurable by systems software so its "personality" can be optimized for a given serial data communications application.

The Z80-SIO is capable of handling asynchronous formats, synchronous byte-oriented protocols such as IBM Bisync, and synchronous bit-oriented protocols such as HDLC and SDLC. This versatile device can also be used to support virtually any other serial protocol for applications other than data communications (cassette or floppy disk interfaces, for example).

The Z80-SIO can generate and check CRC codes in any synchronous mode and can be programmed to check data integrity in various modes. The device also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

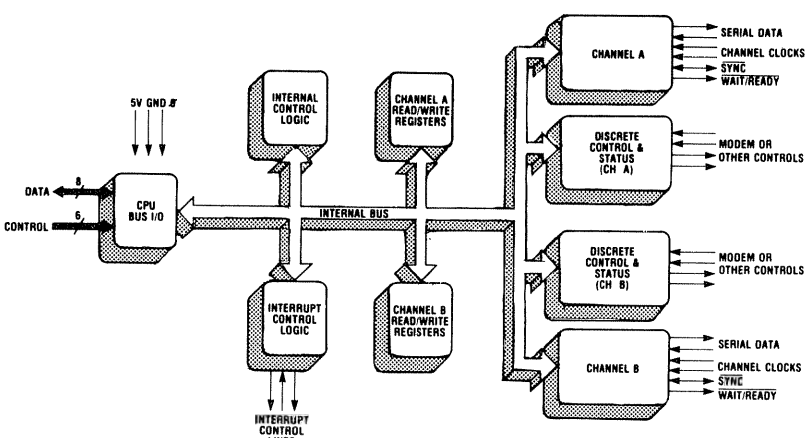
Structure

- N-channel silicon-gate depletion-load technology
- 40-pin DIP
- Single 5 V power supply
- Single-phase 5 V clock
- All inputs and outputs TTL compatible

Features

- Two independent full-duplex channels
- Data rates in synchronous or isosynchronous modes:
  - 0–500K bits/second with 2.5 MHz system clock rate
  - 0–800K bits/second with 4.0 MHz system clock rate
- Receiver data registers quadruply buffered; transmitter doubly buffered.
- Asynchronous features:
  - 5, 6, 7 or 8 bits/character

Fig. 8 - SIO BLOCK DIAGRAM



Z-80 SIO Pin Description

- 1, 1½ or 2 stop bits
- Even, odd or no parity
- ×1, ×16, ×32 and ×64 clock modes
- Break generation and detection
- Parity, overrun and framing error detection

■ Binary synchronous features:
- Internal or external character synchronization
- One or two sync characters in separate registers
- Automatic sync character insertion/deletion
- CRC generation and checking

■ HDLC and SDLC features:
- Abort sequence generation and detection
- Automatic zero insertion and deletion
- Automatic flag insertion between messages
- Address field recognition
- Support for one to eight bits/character
- Valid receive messages protected from overrun
- CRC generation and checking

■ Interrupt features:
- Daisy-chain interrupt logic provides automatic interrupt vectoring with no external logic
- Programmable interrupt vector
- Status Affects Interrupt Vector mode for fast interrupt processing

■ CRC-16 or CRC-CCITT block frame check

■ Separate modem control inputs and outputs for both channels

■ Modem status can be monitored

Pin Description

D_0–D_7. *System Data Bus* (bidirectional, 3-state). The system data bus transfers data and-commands between the CPU and the Z80-SIO. D_0 is the least significant bit.

B/$\overline{A}$. *Channel A Or B Select* (input, High selects Channel B). This input defines which channel is accessed during a data transfer between the CPU and the Z80-SIO. Address bit A_0 from the CPU is often used for the selection function.

C/$\overline{D}$. *Control Or Data Select* (input, High selects Control). This input defines the type of information transfer performed between the CPU and the Z80-SIO. A High at this input during a CPU write to the Z80-SIO causes the information on the data bus to be interpreted as a command for the channel selected by B/$\overline{A}$. A Low at C/$\overline{D}$ means that the information on the data bus is data. Address bit A_1 is often used for this function.

$\overline{CE}$. *Chip Enable* (input, active Low). A Low level at this input enables the Z80-SIO to accept command or data input from the CPU during a write cycle, or to transmit data to the CPU during a read cycle.

ϕ. *System Clock* (input). The Z80-SIO uses the standard Z80 System Clock to synchronize internal signals. This is a single-phase clock.

$\overline{M1}$. *Machine Cycle One* (input from Z80-CPU, active Low). When $\overline{M1}$ is active and $\overline{RD}$ is also active, the Z80-CPU is fetching an instruction from memory; when $\overline{M1}$ is active while $\overline{IORQ}$ is active, the Z80-SIO accepts $\overline{M1}$

FIG. 9 -- Z80-SIO/0 PIN CONFIGURATION

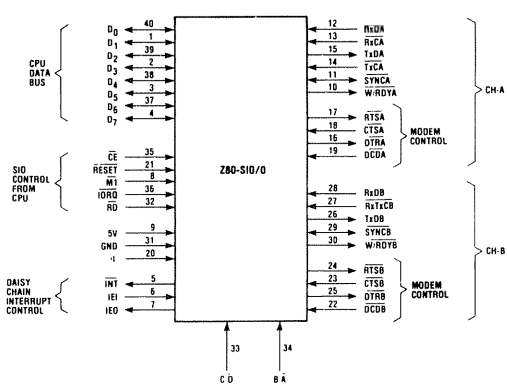
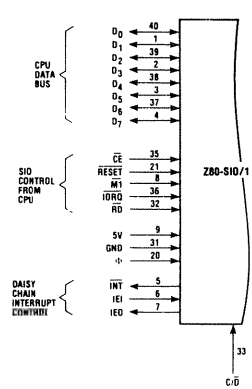


FIG. 10 – Z80-SIO/1 PIN CONFIGURATION



Z-80 SIO Pin Description

and $\overline{IORQ}$ as an interrupt acknowledge if the Z80-SIO is the highest priority device that has interrupted the Z80-CPU.

$\overline{IORQ}$. *Input/Output Request* (input from CPU, active Low). $\overline{IORQ}$ is used in conjunction with B/$\overline{A}$, C/$\overline{D}$, $\overline{CE}$ and $\overline{RD}$ to transfer commands and data between the CPU and the Z80-SIO. When $\overline{CE}$, $\overline{RD}$ and $\overline{IORQ}$ are all active, the channel selected by B/$\overline{A}$ transfers data to the CPU (a read operation). When $\overline{CE}$ and $\overline{IORQ}$ are active, but $\overline{RD}$ is inactive, the channel selected by B/$\overline{A}$ is written to by the CPU with either data or control information as specified by C/$\overline{D}$. As mentioned previously, if $\overline{IORQ}$ and $\overline{M1}$ are active simultaneously, the CPU is acknowledging an interrupt and the Z80-SIO automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

$\overline{RD}$. *Read Cycle Status.* (input from CPU, active Low). If $\overline{RD}$ is active, a memory or I/O read operation is in progress. $\overline{RD}$ is used with B/$\overline{A}$, $\overline{CE}$ and $\overline{IORQ}$ to transfer data from the Z80-SIO to the CPU.

$\overline{RESET}$. *Reset* (input, active Low). A Low $\overline{RESET}$ disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls High and disables all interrupts. The control registers must be rewritten after the Z80-SIO is reset and before data is transmitted or received.

IEI. *Interrupt Enable In* (input, active High). This signal is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

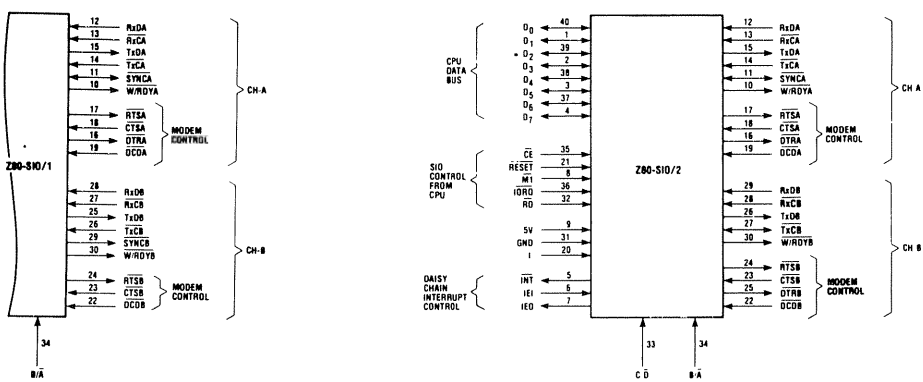
IEO. *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this Z80-SIO. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{INT}$. *Interrupt Request* (output, open drain, active Low). When the Z80-SIO is requesting an interrupt, it pulls $\overline{INT}$ Low.

$\overline{W/RDYA}$, $\overline{W/RDYB}$. *Wait/Ready A, Wait/Ready B* (outputs, open drain when programmed for Wait function, driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the Z80-SIO data rate. The reset state is open drain.

$\overline{CTSA}$, $\overline{CTSB}$. *Clear To Send* (inputs, active Low). When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime signals. The Z80-SIO detects pulses on these inputs and interrupts the CPU on both logic level transitions. The Schmitt-trigger buffering does not guarantee a specified noise-level margin.

FIG. 11 – Z80-SIO/2 PIN CONFIGURATION



Z-80 SIO Bonding Options

$\overline{\text{DCDA}}$, $\overline{\text{DCDB}}$. *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if the Z80-SIO is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow-risetime signals. The Z80-SIO detects pulses on these pins and interrupts the CPU on both logic level transitions. Schmitt-trigger buffering does not guarantee a specific noise level margin.

RxDA, RxDB. *Receive Data* (inputs, active High).

TxDA, TxDB. *Transmit Data* (outputs, active High).

$\overline{\text{RxCA}}$, $\overline{\text{RxCB}}$. *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of RxC. The Receive Clocks may be 1, 16, 32 or 64 times the data rate in Asynchronous modes. These clocks may be driven by the Z80-CTC Counter Timer Circuit for programmable baud rate generation. Both inputs are Schmitt-trigger buffered (no noise level margin is specified). See the following section for bonding options.

$\overline{\text{TxCA}}$, $\overline{\text{TxCB}}$. *Transmitter Clocks* (inputs). TxD changes on the falling edge of $\overline{\text{TxC}}$. In Asynchronous modes, the Transmitter Clocks may be 1, 16, 32 or 64 times the data rate; however, the clock multiplier for the transmitter and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements (no noise level margin is specified). Transmitter Clocks may be driven by the Z80-CTC Counter Timer Circuit for programmable baud rate generation. See the following section for bonding options.

$\overline{\text{RTSA}}$, $\overline{\text{RTSB}}$. *Request To Send* (outputs, active Low). When the RTS bit is set, the $\overline{\text{RTS}}$ output goes Low. When the RTS bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the $\overline{\text{RTS}}$ pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

$\overline{\text{DTRA}}$, $\overline{\text{DTRB}}$. *Data Terminal Ready* (outputs, active Low). See note on bonding options. These outputs follow the state programmed into the DTR bit. They can also be programmed as general-purpose outputs.

$\overline{\text{SYNC}}$ A, $\overline{\text{SYNC}}$ B. *Synchronization* (inputs/outputs, active Low). These pins can act either as inputs or outputs. In the Asynchronous Receive mode, they are inputs similar to $\overline{\text{CTS}}$ and $\overline{\text{DCD}}$. In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in RR0. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved, $\overline{\text{SYNC}}$ must be driven Low on the second rising edge of $\overline{\text{RxC}}$ after that rising edge of $\overline{\text{RxC}}$ on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the $\overline{\text{SYNC}}$ input. Once $\overline{\text{SYNC}}$ is forced Low, it is wise to keep it Low until the CPU informs the external sync logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of $\overline{\text{RxC}}$ that immediately precedes the falling edge of $\overline{\text{SYNC}}$ in the External Sync mode.

In the Internal Synchronization mode (Monosync and Bisync), these pins act as outputs that are active during the part of the receive clock ($\overline{\text{RxC}}$) cycle in which sync characters are recognized. The sync condition is not latched, so these outputs are active each time a sync pattern is recognized, regardless of character boundaries.

Bonding Options

The constraints of a 40-pin package make it impossible to bring out the Receive Clock, Transmit Clock, Data Terminal Ready and Sync signals for both channels. Therefore, Channel B must sacrifice a signal or have two signals bonded together. Since user requirements vary, three bonding options are offered:

- Z80-SIO/0 has all four signals, but $\overline{\text{TxCB}}$ and $\overline{\text{RxCB}}$ are bonded together (Fig. 9).
- Z80-SIO/1 sacrifices $\overline{\text{DTRB}}$ and keeps $\overline{\text{TxCB}}$, $\overline{\text{RxCB}}$ and $\overline{\text{SYNCB}}$ (Fig. 10).
- Z80-SIO/2 sacrifices $\overline{\text{SYNCB}}$ and keeps $\overline{\text{TxCB}}$, $\overline{\text{RxCB}}$ and $\overline{\text{DTRB}}$ (Fig. 11).

Architecture

The device internal structure includes a Z80-CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains read and write registers, and discrete control and status logic that provides the interface to modems or other external devices.

The read and write register group includes five 8-bit control registers, two sync-character registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B that may be read through Read Register 2 in Channel B. The registers for both channels are designated in the text as follows:

WR0-WR7 — Write Registers 0 through 7
RR0-RR2 — Read Registers 0 through 2

The bit assignment and functional grouping of each register is configured to simplify and organize the programming process. Table 1 lists the functions assigned to each read or write register.

RR0	Transmit/Receive buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)

Read Register Functions

Z-80 SIO Architecture

WR0	Register pointers, CRC initialize, initialization commands for the various modes, etc.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync character or SDLC address field
WR7	Sync character or SDLC flag

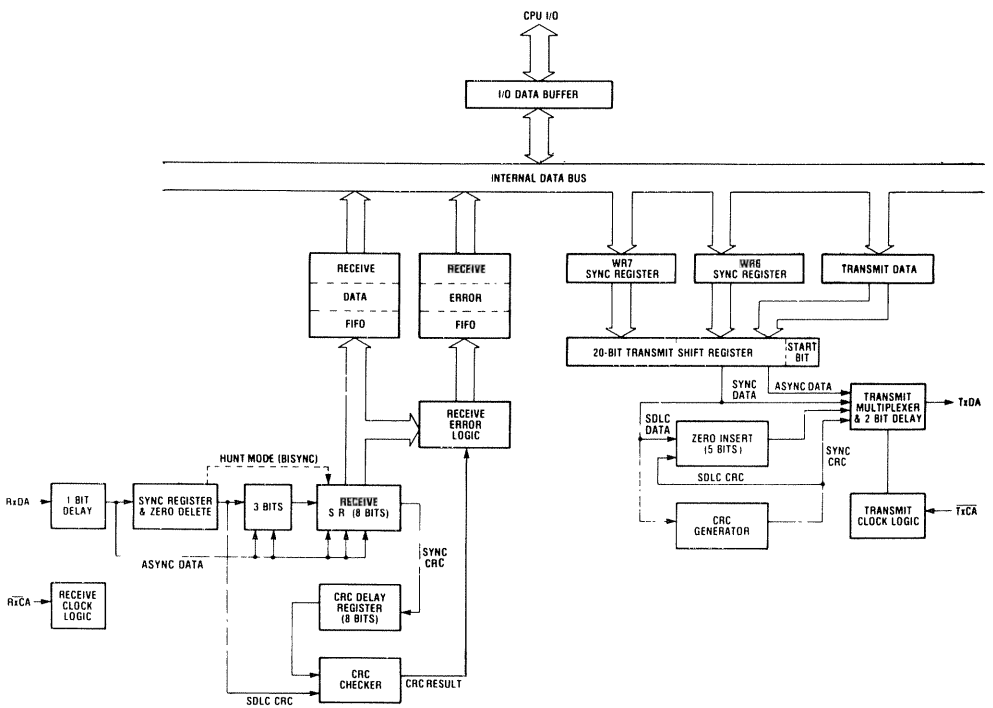
Write Register Functions

Table 1. Functional Assignments of Read and Write Registers

The logic for both channels provides formats, synchronization and validation for data transferred to and from the channel interface. The modem control inputs Clear to Send ($\overline{CTS}$) and Data Carrier Detect ($\overline{DCD}$) are monitored by the discrete control logic under program control. All the modem control signals are general purpose in nature and can be used for functions other than modem control.

For automatic interrupt vectoring, the interrupt control logic determines which channel and which device within the channel has the highest priority. Priority is fixed with Channel A assigned a higher priority than Channel B; Receive, Transmit and External/Status interrupts are prioritized in that order within each channel.

FIG. 12 – TRANSMIT AND RECEIVE DATA PATH



Z-80 SIO Architecture

Data Path

The transmit and receive data path illustrated for Channel A in Figure 12 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode and—in Asynchronous modes—the character length.

The transmitter has an 8-bit transmit data register that is loaded from the internal data bus, and a 20-bit transmit shift register that can be loaded from the sync character buffers (WR6 and WR7) or from the transmit data register. Depending on the operational mode, outgoing data is routed through one of four main paths before it is transmitted from the Transmit Data Output (TxD).

Functional Description

The functional capabilities of the Z80-SIO can be described from two different points of view: as a data communications device, it transmits and receives serial data, and meets the requirements of various data communications protocols; as a Z80 family peripheral, it interacts with the Z80-CPU and other Z80 peripheral circuits, and shares the data, address and control busses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessors, the Z80-SIO offers valuable features such as non-vectored interrupts, polling and simple handshake capability.

The first part of the following functional description describes the interaction between the CPU and Z80-SIO; the second part introduces its data communications capabilities.

I/O Interface Capabilities

The Z80-SIO offers the choice of Polling, Interrupt (vectored or non-vectored) and Block Transfer modes to transfer data, status and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

Polling. There are no interrupts in the Polled mode. Status registers RR0 and RR1 are updated at appropriate times for each function being performed (for example, CRC Error status valid at the end of the message). All the interrupt modes of the Z80-SIO must be disabled to operate the device in a polled environment.

While in its Polling sequence, the CPU examines the status contained in RR0 for each channel; the RR0 status bits serve as an acknowledge to the Poll inquiry. The two RR0 status bits D_0 and D_2 indicate that a data transfer is needed. The status also indicates Error or other special status conditions (see "Z80-SIO Programming"). The Special Receive Condition status contained in RR1 does not have to be read in a Polling sequence because the status bits in RR1 must be accompanied by a Receive Character Available status in RR0.

Interrupts. The Z80-SIO offers an elaborate interrupt scheme to provide fast interrupt response in real-time applications. Channel B registers WR2 and RR2 contain the interrupt vector that points to an interrupt service routine in the memory. To service operations in both channels and to eliminate the necessity of writing a status analysis routine, the Z80-SIO can modify the interrupt vector in RR2 so it points directly to one of eight interrupt service routines. This is done under program control by setting a program bit (WR1, D_2) in Channel B called "Status Affects Vector." When this bit is set, the interrupt vector in WR2 is modified according to the assigned priority of the various interrupting conditions. The table in the Write Register 1 description (Z80-SIO Programming section) shows the modification details.

Transmit interrupts, Receive interrupts and External/Status interrupts are the main sources of interrupts. Each interrupt source is enabled under program control with Channel A having a higher priority than Channel B, and with Receiver, Transmit and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted by the transmit buffer *becoming* empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways:

- Interrupt on the first received character
- Interrupt on all received characters
- Interrupt on a Special Receive condition

Interrupt On First Character is typically used with the Block Transfer mode. Interrupt On All Receive Characters has the option of modifying the interrupt vector in the event of a parity error. The Special Receive Condition interrupt can occur on a character or message basis (End Of Frame interrupt in SDLC, for example). The Special Receive condition can cause an interrupt only if the Interrupt On First Receive Character or Interrupt On All Receive Characters mode is selected. In Interrupt On First Receive Character, an interrupt can occur from Special Receive conditions (except Parity Error) after the first receive character interrupt (example: Receive Overrun interrupt).

The main function of the External/Status interrupt is to monitor the signal transitions of the $\overline{CTS}$, $\overline{DCD}$ and $\overline{SYNC}$ pins; however, an External/Status interrupt is also caused by a Transmit Underrun condition or by the detection of a Break (Asynchronous mode) or Abort (SDLC mode) sequence in the data stream. The interrupt caused by the Break/Abort sequence has a special feature that allows the Z80-SIO to interrupt when the Break/Abort sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and

Functional Description

the accurate timing of the Break/Abort condition in external logic.

CPU/DMA Block Transfer. The Z80-SIO provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers (Z80-DMA or other designs). The Block Transfer mode uses the $\overline{\text{WAIT/READY}}$ output in conjunction with the Wait/Ready bits of Write Register 1. The $\overline{\text{WAIT/READY}}$ output can be defined under software control as a $\overline{\text{WAIT}}$ line in the CPU Block Transfer mode or as a $\overline{\text{READY}}$ line in the DMA Block Transfer mode.

To a DMA controller, the Z80-SIO READY output indicates that the Z80-SIO is ready to transfer data to or from memory. To the CPU, the WAIT output indicates that the Z80-SIO is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle. The programming of bits 5, 6 and 7 of Write Register 1 and the logic states of the $\overline{\text{WAIT/READY}}$ line are defined in the Write Register 1 description (Z80-SIO Programming section).

Data Communications Capabilities

In addition to the I/O capabilities previously discussed, the Z80-SIO provides two independent full-duplex channels that can be programmed for use in Asynchronous, Synchronous and SDLC (HDLC) modes. These different modes are provided to facilitate the implementation of commonly used data communications protocols. The following is a short description of the data communications protocols supported by the Z80-SIO. A more detailed explanation of these modes can be found in the *Z80-SIO Technical Manual*.

Asynchronous Modes. The Z80-SIO offers transmission and reception of five to eight bits per character, plus optional even or odd parity. The transmitter can supply one, one and a half or two stop bits per character and can provide a break output at any time. The receiver break detection logic interrupts the CPU only at the start and end of a received break. Reception is protected from spikes by a transient spike rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the Receive Data input. If the Low does not persist—as in the case of a transient—the character assembly process is not started.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occurred. Vectored interrupts allow fast servicing of error conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The Z80-SIO does not require symmetric Transmit and Receive Clock signals—a feature that allows it to be used with a Z80-CTC or any other clock source. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32 or 1/64 of the clock rate supplied to the Receive and Transmit Clock inputs.

In Asynchronous modes, the $\overline{\text{SYNC}}$ pin may be programmed for an input that can be used for functions such as monitoring a ring indicator.

Synchronous Modes. The Z80-SIO supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols can be handled in several modes that allow character synchronization with an 8-bit sync character (Monosync), any 16-bit sync pattern (Bisync), or with an external sync signal. Leading sync characters can be removed without interrupting the CPU. CRC checking for synchronous byte-oriented modes is delayed by one character time so the CPU may disable CRC checking on specific characters. This permits implementation of protocols such as IBM Bisync.

Both CRC-16 ($X^{16}+X^{15}+X^2+1$) and CCITT ($X^{16}+X^{12}+X^5+1$) error checking polynomials are supported. In all non-SDLC modes, the CRC generator is initialized to 0's; in SDLC modes, it is initialized to 1's. (This means that the Z80-SIO cannot generate or check CRC for IBM-compatible soft-sectored disks.) The Z80-SIO also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows very high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 8- or 16-bit sync characters regardless of the programmed character length. Since the CPU can read status information from the Z80-SIO, it can determine the type of transmission (data, CRC or sync characters) that is taking place at any time.

The Z80-SIO supports synchronous bit-oriented protocols such as SDLC and HDLC by performing automatic flag sending, zero insertion and CRC generation. A special command can be used to abort a frame in transmission. The Z80-SIO automatically transmits the CRC and trailing flag when the transmit buffer becomes empty. An interrupt warns the CPU of this status change so an abort may be issued if a transmitter underrun has occurred. One to eight bits per character can be sent, which allows transmission of a message exactly as received with no prior information about the character structure in the information field of a frame.

The receiver automatically synchronizes on the leading flag of a frame and provides a synchronization signal that can be programmed to interrupt. In addition, an interrupt on the first received character or on every character can be selected. The receiver automatically deletes all zeroes inserted by the transmitter during character assembly. It also calculates and automatically checks the CRC to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. The receiver can be programmed to search for frames addressed to only a specified user-selectable address or to a global broadcast address. In this mode, frames that do not match the user-

selected or broadcast address are ignored. The Address Search mode provides for a single-byte address recognizable by the hardware. The number of address bytes can be extended under software control.

The Z80-SIO can be conveniently used under DMA control to provide high-speed reception. The Z80-SIO can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The Z80-SIO then issues an End Of Frame interrupt and the CPU checks the status of the received message. Thus, the CPU is freed for other service while the message is being received. A similar scheme allows message transmission under DMA control.

Z80-SIO Programming

To program the Z80-SIO, the system program first issues a series of commands that initialize the basic mode of operation and then other commands that qualify conditions within the selected mode. For example, the Asynchronous mode, character length, clock rate, number of stop bits, even or odd parity are first set, then the interrupt mode and, finally, receiver or transmitter enable. The WR4 parameters must be issued before any other parameters are issued in the initialization routine.

Both channels contain command registers that must be programmed via the system program prior to operation. The Channel Select input (B/$\overline{A}$) and the Control/Data input (C/$\overline{D}$) are the command structure addressing controls, and are normally controlled by the CPU address bus. Figure 14 illustrates the timing relationships for programming the write registers, and transferring data and status.

Write Registers

The Z80-SIO contains eight registers (WR0-WR7) in each channel that are programmed separately by the system program to configure the functional personality of the channels. With the exception of WR0, programming the write registers requires two bytes. The first byte contains three bits (D_0-D_2) that point to the selected register; the second byte is the actual control word that is written into the register to configure the Z80-SIO.

WR0 is a special case in that all the basic commands (CMD_0-CMD_2) can be accessed with a single byte. Reset (internal or external) initializes the pointer bits D_0-D_2 to point to WR0.

Read Registers

The Z80-SIO contains three registers, RR0-RR2 (Figure 12).that can be read to obtain the status information for each channel (except for RR2 — Channel B only). The status information includes error conditions, interrupt vector and standard communications-interface signals.

To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing an input instruction, the contents of the addressed read register can be read by the CPU.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Condition interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

FIG. 12 – READ REGISTER BIT FUNTIONS

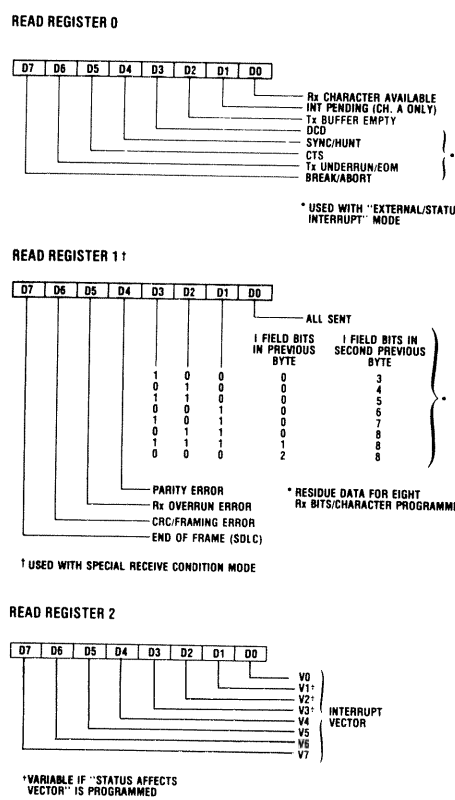
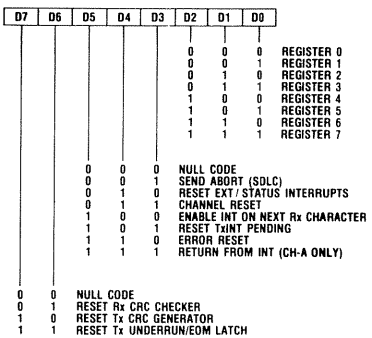


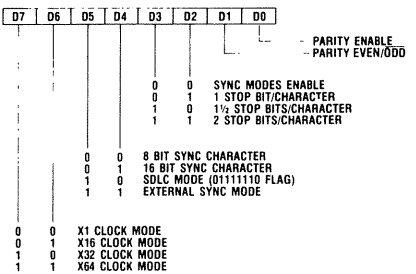
FIG. 13 – WRITE REGISTER BIT FUNCTIONS

Z-80 SIO Programming

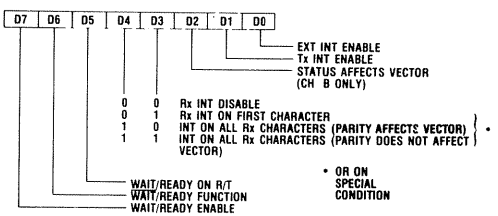
WRITE REGISTER 0



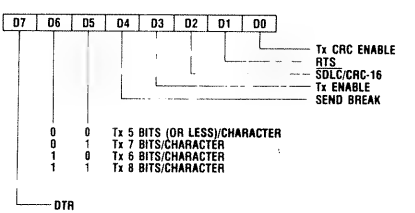
WRITE REGISTER 4



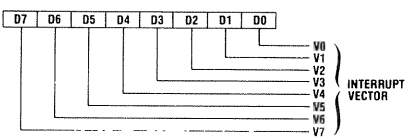
WRITE REGISTER 1



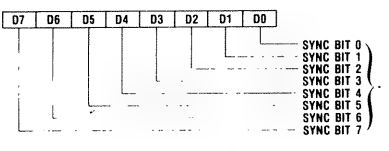
WRITE REGISTER 5



WRITE REGISTER 2 (CHANNEL B ONLY)

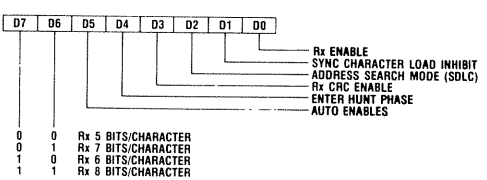


WRITE REGISTER 6

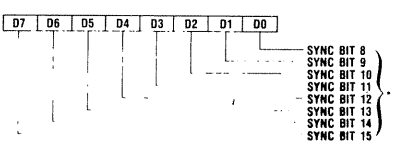


*ALSO SDLC ADDRESS FIELD

WRITE REGISTER 3



WRITE REGISTER 7



*FOR SDLC IT MUST BE PROGRAMMED TO "01111110" FOR FLAG RECOGNITION

Z-80 SIO Timing

Timing

Read Cycle. The timing signals generated by a Z80-CPU input instruction to read a Data or Status byte from the Z80-SIO are illustrated in Figure 14a.

Write Cycle. Figure 14b illustrates the timing and data signals generated by a Z80-CPU output instruction to write a Data or Control byte into the Z80-SIO.

FIG. 14a – READ CYCLE

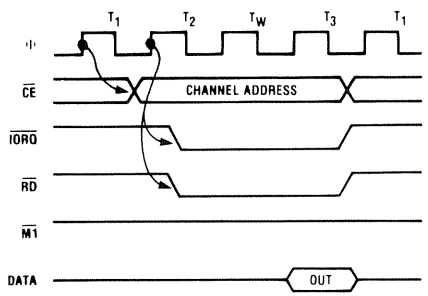
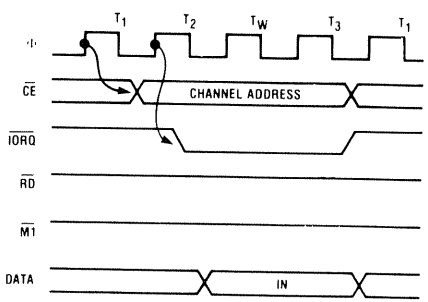


FIG. 14b – WRITE CYCLE



Interrupt Acknowledge Cycle. After receiving an Interrupt Request signal ($\overline{INT}$ pulled Low), the Z80-CPU sends an Interrupt Acknowledge signal ($\overline{M1}$ and $\overline{IORQ}$ both Low). The daisy-chained interrupt circuits determine the highest priority interrupt requestor. The IEI of the highest priority peripheral is terminated High. For any peripheral that has no interrupt pending or under service, IEO = IEI. Any peripheral that does have an interrupt pending or under service forces its IEO Low.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while $\overline{M1}$ is Low. When $\overline{IORQ}$ is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

Return From Interrupt Cycle. Normally, the Z80-CPU issues a RETI (RETurn from Interrupt) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch to terminate the interrupt that has just been processed. This is accomplished by manipulating the daisy chain in the following way.

The normal daisy chain operation can be used to detect a pending interrupt; however, it cannot distinguish between an interrupt under service and a pending unacknowledged interrupt of a higher priority. Whenever "ED" is decoded, the daisy chain is modified by forcing High the IEO of any interrupt that has not yet been acknowledged. Thus the daisy chain identifies the device presently under service as the only one with an IEI

FIG. 14c – INTERRUPT ACKNOWLEDGE CYCLE

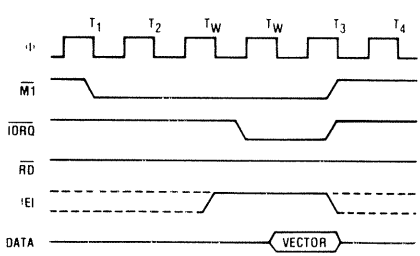
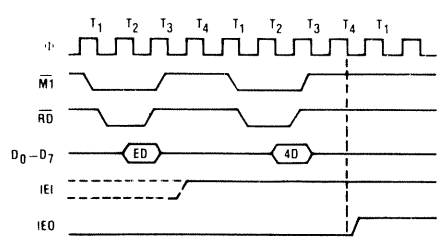


FIG. 14d – RETURN FROM INTERRUPT CYCLE



Z-80 SIO Timing

High and an IEO Low. If the next opcode byte is "4D," the interrupt-under-service latch is reset.

The ripple time of the interrupt daisy chain (both the High-to-Low and the Low-to-High transitions) limits the number of devices that can be placed in the daisy chain. Ripple time can be improved with carry-look-ahead, or by extending the interrupt acknowledge cycle.

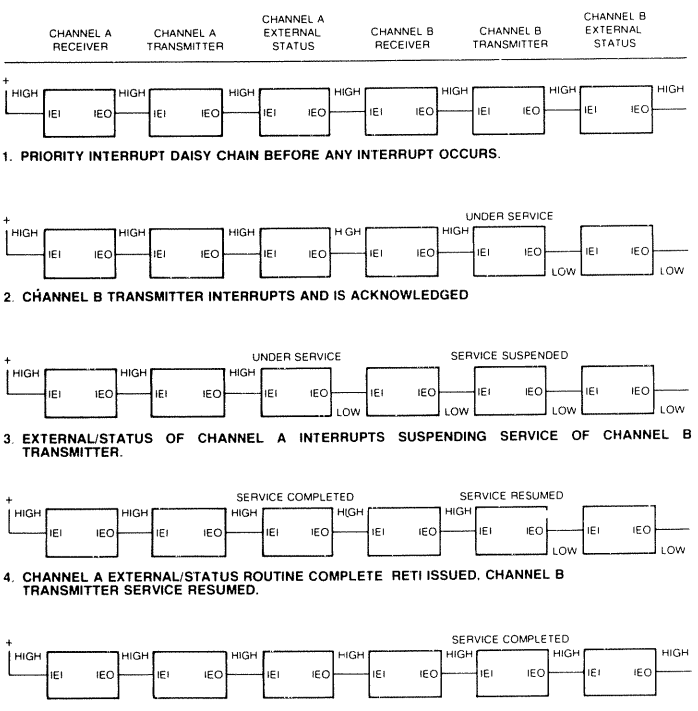
Daisy Chain Interrupt Nesting

Figure 15 illustrates the daisy chain configuration of interrupt circuits and their behavior with nested interrupts (an interrupt that is interrupted by another with a higher priority).

Each box in the illustration could be a separate external Z80 peripheral circuit with a user-defined order of interrupt priorities. However, a similar daisy chain structure also exists inside the Z80-SIO, which has six interrupt levels with a fixed order of priorities.

The case illustrated occurs when the transmitter of Channel B interrupts and is granted service. While this interrupt is being serviced, it is interrupted by a higher priority interrupt from Channel A. The second interrupt is serviced and—upon completion—a RETI instruction is executed or a RETI command is written into the Z80-SIO, resetting the interrupt-under-service latch of the Channel A interrupt. At this time, the service rotine for Channel B is resumed. When it is completed, another RETI instruction is executed to complete the interrupt service.

FIG. 15 – TYPICAL INTERRUPT SEQUENCE



1. PRIORITY INTERRUPT DAISY CHAIN BEFORE ANY INTERRUPT OCCURS.

2. CHANNEL B TRANSMITTER INTERRUPTS AND IS ACKNOWLEDGED

3. EXTERNAL/STATUS OF CHANNEL A INTERRUPTS SUSPENDING SERVICE OF CHANNEL B TRANSMITTER.

4. CHANNEL A EXTERNAL/STATUS ROUTINE COMPLETE RETI ISSUED, CHANNEL B TRANSMITTER SERVICE RESUMED.

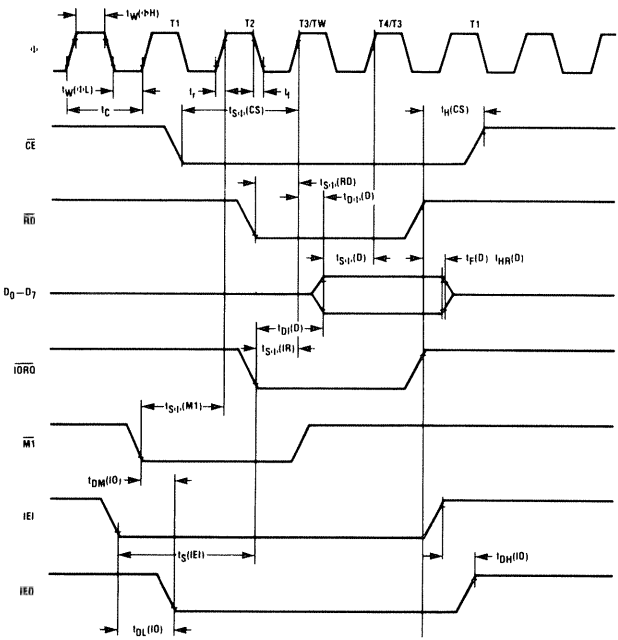
5. CHANNEL B TRANSMITTER SERVICE ROUTINE COMPLETE, SECOND RETI ISSUED.

Electrical Characteristics

FIG. 16 – TYPICAL INTERRUPT SEQUENCE

AC Characteristics

$T_A = 0°C$, $V_{CC} = +5V$, $\pm 5\%$

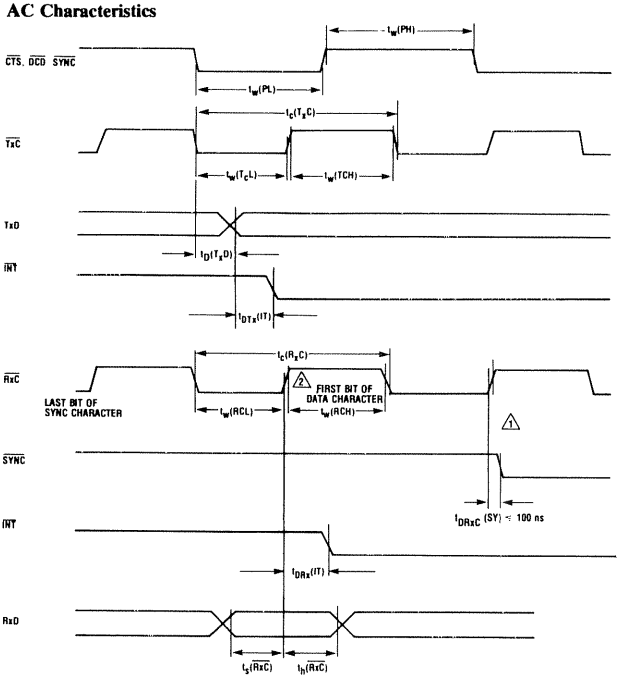


Signal	Symbol	Parameter	Z80-SIO Min	Z80-SIO Max	Z80A-SIO Min	Z80A-SIO Max	Unit
φ	$t_{c(\phi)}$	Clock Period	400	4000	250	4000	ns
	$t_{w(\phi H)}$	Clock Pulse Width clock HIGH	170	2000	105	2000	ns
	$t_{w(\phi L)}$	Clock Pulse Width clock LOW	170	2000	105	2000	ns
	t_r, t_f	Clock Rise and Fall Times	0	30	0	30	ns
	t_m	Any Unspecified Hold Time for setup times specified below	0		0		ns
$\overline{CE}$ $\overline{B/A}$ $C/\overline{D}$ $\overline{IORQ}$	$t_{s(CS)}$	Control Signal Setup Time to rising edge of φ during Read or Write Cycle	160		145		ns
D_0–D_7	$t_{d(D)}$	Data Output Delay from rising edge of φ during Read Cycle		240		220	ns
	$t_{s(D)}$	Data Setup Time to rising edge of φ during Write or M1 Cycle	50		50		ns
	$t_{df(D)}$	Data Output Delay from falling edge of $\overline{IORQ}$ during INTA Cycle		340		160	ns
	$t_f(D)$	Delay to Floating Bus (output buffer disable time)		230		110	ns
IEI	$t_s(IEI)$	IEI Setup Time to falling edge of $\overline{IORQ}$ during INTA Cycle	200		140		ns
IEO	$t_{dm(IO)}$	IEO Delay Time from rising edge of IEI (after ED decode)		150		100	ns
	$t_{df(IO)}$	IEO Delay Time from falling edge of IEI		150		100	ns
	$t_{dM(IO)}$	IEO Delay Time from falling edge of $\overline{M1}$ (interrupt occurring just prior to M1)		300		190	ns
M1	$t_{s(M1)}$	$\overline{M1}$ Setup Time to rising edge of φ during INTA or M1 Cycle	210		90		ns
$\overline{RD}$	$t_{s(RD)}$	$\overline{RD}$ Setup Time to rising edge of φ during Read or M1 Cycle	240		115		ns

*If WAIT from the SIO is to be used, $\overline{CE}$, $\overline{IORQ}$, $C/\overline{D}$ and $\overline{M1}$ must be valid for as long as the Wait condition is to persist.

Electrical Characteristics

AC Characteristics



NOTES:

1 The SYNC input must be driven Low on the rising edge of RxC delayed two complete clock cycles from the last bit of the sync character

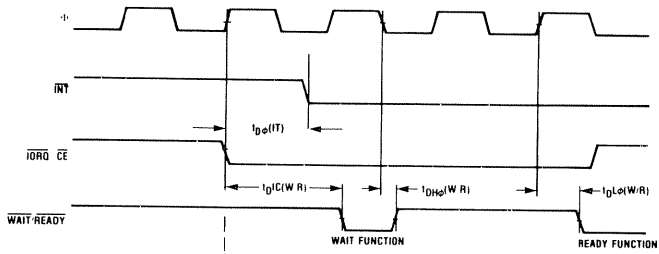
2 Data character assembly begins on the next Receive Clock cycle after the last bit of the sync character is received

Signal	Symbol	Parameter	Z80-SIO Min	Z80-SIO Max	Z80A-SIO Min	Z80A-SIO Max	Unit
INT	t_{DH}(IT)	INT Delay Time from rising edge of RxC	10	13	10	13	φ periods
	t_{Df}(IT)	INT Delay Time from transition of Xmit Data Bit	5	9	5	9	φ periods
CTSA CTSB, DCDA, DCDB, SYNCA SYNCB	t_w(PH)	Minimum HIGH Pulse Width for latching state into register and generating interrupt	200		200		ns
	t_w(PL)	Minimum LOW Pulse Width for latching state into register and generating interrupt	200		200		ns
SYNCA SYNCB	t_{DL}(SY)	Sync Pulse Delay Time from rising edge of RxC Output Modes	4	7	4	7	φ periods
	$t_{DR×C}$(SY)	Sync Pulse Delay Time from rising edge of RxC External Sync Mode		100		100	ns
TxCA TxCB	t_c(TxC)	Transmit Clock Period	400	∞	400	∞	ns
	t_w(TCH)	Transmit Clock Pulse Width clock HIGH	180	∞	180	∞	ns
	t_w(TCL)	Transmit Clock Pulse Width clock LOW	180	∞	180	∞	ns
TxDA TxDB†	t_D(TxD)	TxD Output Delay from falling Edge of TxC (x1 Clock Mode)		400		300	ns
RxCA RxCB	t_c(RxC)	Receive Clock Period	400	∞	400	∞	ns
	t_w(RCH)	Receive Clock Pulse Width clock HIGH	180	∞	180	∞	ns
	t_w(RCL)	Receive Clock Pulse Width clock LOW	180	∞	180	∞	ns
RxDA RxDB†	t_s(RxC)	Setup Time to rising edge of RxC x1 mode	0		0		ns
	t_h(RxC)	Hold Time from rising edge of RxC x1 mode	140		140		ns

†In all modes, the system clock (φ) rate must be at least 4 5 times the maximum data rate. RESET must be active a minimum of one complete φ cycle

Electrical Characteristics

AC Characteristics



Signal	Symbol	Parameter	Z80-SIO Min	Z80-SIO Max	Z80A-SIO Min	Z80A-SIO Max	Unit
INT	$t_{\phi(IT)}$	INT Delay Time from rising edge of ϕ		200		200	ns
WAIT/READY	$t_{DIC(W/R)}$	WAIT/READY Delay Time from IORQ or CE in Wait Mode		300		210	ns
	$t_{DH\phi(W/R)}$	WAIT/READY Delay Time from falling edge of ϕ WAIT/READY HIGH Wait Mode		150		130	ns
	$t_{DRx(W/R)}$	WAIT/READY Delay Time from rising edge of RxC Data Bit, Ready Mode	10	13	10	13	ϕ periods
	$t_{DTx(W/R)}$	WAIT/READY Delay Time from center of Transmit Data Bit, Ready Mode	5	9	5	9	ϕ periods
	$t_{DL\phi(W/R)}$	WAIT/READY Delay Time from rising edge of ϕ WAIT/READY LOW Ready Mode		120		120	ns

DC Characteristics

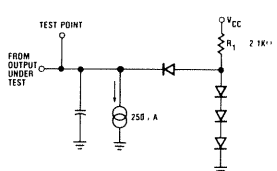
$T_A = 0\,°C$ to $70\,°C$, $V_{CC} = +5V$, $\pm 5\%$

Symbol	Parameter	Min	Max	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	−0.3	+0.45	V	
V_{IHC}	Clock Input High Voltage	$V_{CC}-0.6$	+5.5	V	
V_{IL}	Input Low Voltage	−0.3	+0.8	V	
V_{IH}	Input High Voltage	+2.0	+5.5	V	
V_{OL}	Output Low Voltage		+0.4	V	$I_{OL} = 2.0$ mA
V_{OH}	Output High Voltage	+2.4		V	$I_{OH} = -250\,\mu A$
I_{LI}	Input Leakage Current	−10	+10	μA	$0 \leq V_{IN} \leq V_{CC}$
I_Z	3-State Output/Data Bus Input Leakage Current	−10	+10	μA	$0 \leq V_{IN} \leq V_{CC}$
I_{LSYI}	SYNC Pin Leakage Current	−40	+10	μA	$0 \leq V_{IN} \leq V_{CC}$
I_{CC}	Power Supply Current		100	mA	

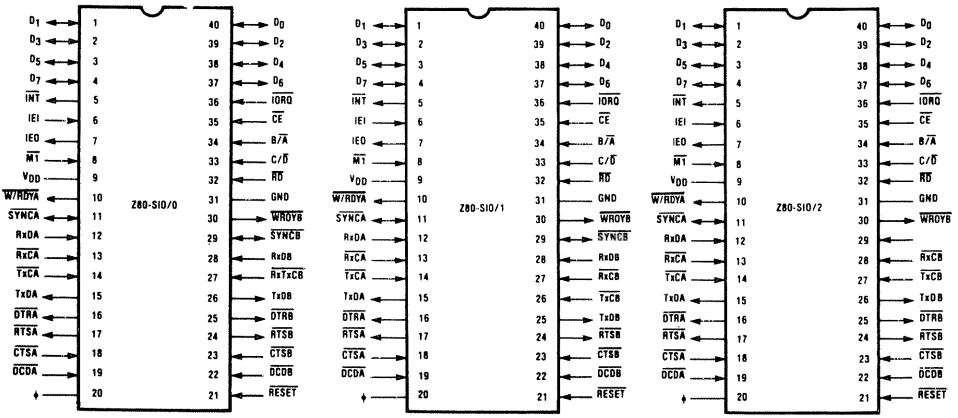
Capacitance

$T_A = 25\,°C$, $f = 1$ MHz

Symbol	Parameter	Min	Max	Unit	Test Condition
C	Clock Capacitance		40	pF	Unmeasured pins returned to ground
C_{IN}	Input Capacitance		5	pF	
C_{OUT}	Output Capacitance		10	pF	

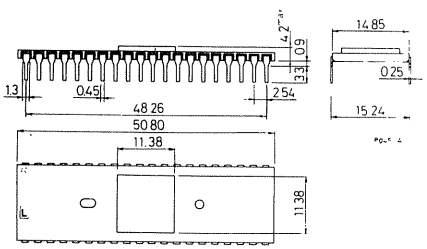


Package Information

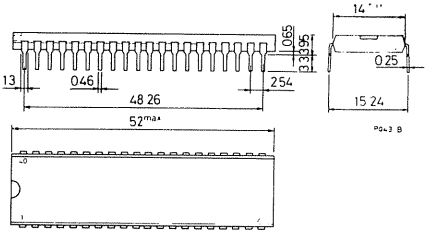


MECHANICAL DATA (dimensions in mm)

40-PIN CERAMIC DUAL IN-LINE SLAM PACKAGE



40-PIN PLASTIC DUAL IN-LINE PACKAGE



ORDERING NUMBERS:

Z80-SIO D1 for dual in-line ceramic slam package
Z80-SIO B1 for dual in-line plastic package
Z80A-SIO D1 for dual in-line ceramic slam package
Z80A-SIO B1 for dual in-line plastic package

Z 80-SIO/9
Z 80A-SIO/9

Product Specification

General Description

The Z80-SIO (Serial Input/Output) is a single-channel multi-function peripheral component designed to satisfy a wide variety of serial data communications requirements in microcomputer systems. Its basic function is a serial-to-parallel, parallel-to-serial converter/controller, but—within that role—it is configurable by systems software so its "personality" can be optimized for a given serial data communications application.

The Z80-SIO is capable of handling asynchronous formats, synchronous byte-oriented protocols such as IBM Bisync, and synchronous bit-oriented protocols such as HDLC and SDLC. This versatile device can also be used to support virtually any other serial protocol for applications other than data communications (cassette or floppy disk interfaces, for example).

The Z80-SIO can generate and check CRC codes in any synchronous mode and can be programmed to check data integrity in various modes. The device also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

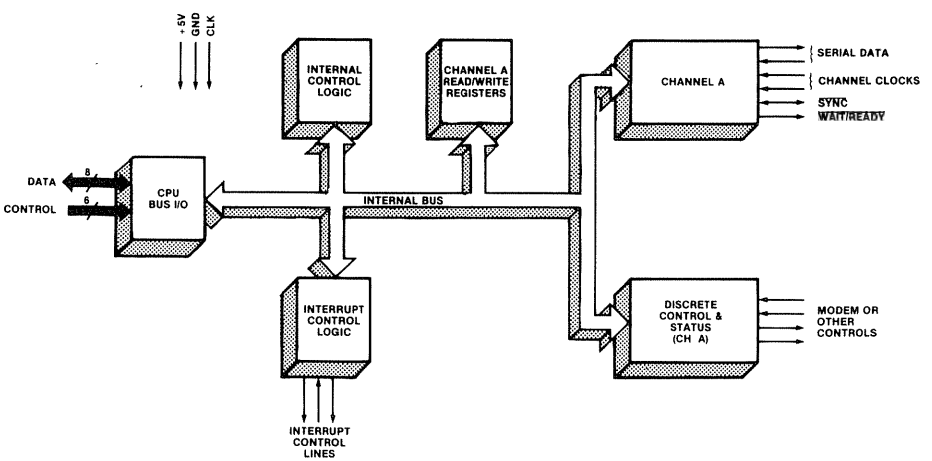
Structure

- N-channel silicon-gate depletion-load technology
- 40-pin DIP
- Single 5V power supply
- Single-phase 5V clock
- All inputs and outputs TTL compatible

Features

- One full-duplex channel
- Data rates in synchronous or isosynchronous modes:
  - 0–500K bits/second with 2.5 MHz system clock rate
  - 0–800K bits/second with 4.0 MHz system clock rate
- Receiver data registers quadruply buffered; transmitter doubly buffered.
- Asynchronous features:
  - 5, 6, 7 or 8 bits/character
  - 1, 1½ or 2 stop bits
  - Even, odd or no parity

FIG. 17 – SIO/9 BLOCK DIAGRAM



- × 1, × 16, × 32 and × 64 clock modes
- Break generation and detection
- Parity, overrun and framing error detection

■ Binary synchronous features:
- Internal or external character synchronization
- One or two sync characters in separate registers
- Automatic sync character insertion/deletion
- CRC generation and checking

■ HDLC and SDLC features:
- Abort sequence generation and detection
- Automatic zero insertion and deletion
- Automatic flag insertion between messages
- Address field recognition
- Support for one to eight bits/character
- Valid receive messages protected from overrun
- CRC generation and checking

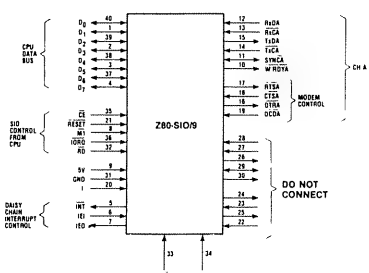
■ Interrupt features:
- Daisy-chain interrupt logic provides automatic interrupt vectoring with no external logic
- Programmable interrupt vector
- Status Affects Interrupt Vector mode for fast interrupt processing

■ CRC-16 or CRC-CCITT block frame check

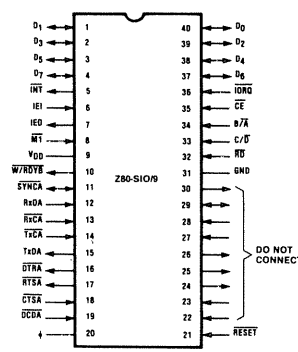
■ Modem control inputs and outputs

■ Modem status can be monitored

Write Register 2 (interrupt vector) and the Status Affects Vector bit in Write Register 1 are, however, still programmed by selecting Channel B with the B/A select input. All other bits in Write Register 1 or Channel B must be programmed to 0.



FUNCTIONAL PIN CONFIGURATION



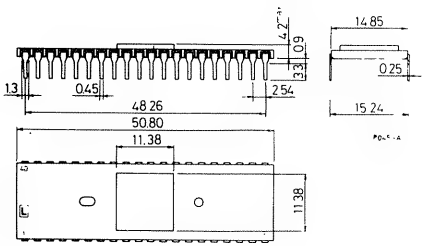
PACKAGE CONFIGURATION

Ordering Information

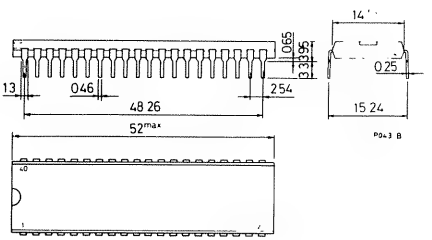
Z80A-SIO/9 D1	= 4 MHz maximum clock rate, ceramic package, 0 °C to + 70 °C temperature range	Z80-SIO/9 D1	= 2.5 MHz maximum clock rate, ceramic package, 0 °C to + 70 °C temperature range
Z80A-SIO/9 B1	= 4 MHz maximum clock rate, plastic package, 0 °C to + 70 °C temperature range	Z80-SIO/9 B1	= 2.5 MHz maximum clock rate, plastic package, 0 °C to + 70 °C temperature range

MECHANICAL DATA (dimensions in mm)

40-PIN CERAMIC DUAL IN-LINE SLAM PACKAGE



40-PIN PLASTIC DUAL IN-LINE PACKAGE



ORDERING NUMBERS:

Z80-SIO/9 D1 for dual in-line ceramic slam package
Z80-SIO/9 B1 for dual in-line plastic package
Z80A-SIO/9 D1 for dual in-line ceramic slam package
Z80A-SIO/9 B1 for dual in-line plastic package

Z 80-DART
Z 80A-DART

Product Specification

Features

- Two independent full-duplex channels with separate modem controls. Modem status can be monitored.
- Receiver data registers are quadruply buffered; the transmitter is doubly buffered.
- Interrupt features include a programmable interrupt vector, a "status affects vector" mode for fast interrupt processing, and the standard Z80 peripheral daisy-chain interrupt structure that provides automatic interrrupt vectoring with no external logic.
- In x1 clock mode, data rates are 0 to 500K bits/second with a 2.5 MHz clock, or 0 to 800K bits/second with a 4.0 MHz clock.
- Programmable options include 1, 1 $\frac{1}{2}$ or 2 stop bits; even, odd or no parity; and x1, x16, x32 and x64 clock modes.
- Break generation and detection as well as parity-, overrun-and framing-error detection are available.

Description

The Z80 DART (Dual-Channel Asynchronous Receiver/Transmitter) is a dual-channel multi-function peripheral component that satisfies a wide variety of asynchronous serial data communications requirements in micro-computer systems. The Z80 DART is used as a serial-to-parallel, parallel-to-serial converter/controller in asynchronous applications. In addition, the device also provides modem controls for both channel. In application where modem controls are not needed, these lines can be used for general-purpose I/O.

SGS-ATES also offers the Z80 SIO, a more versatile device that provides synchronous (Bisync, HDLC and SDLC) as well as asynchronous operation).

The Z80 DART is fabricated with n-channel silicon-gate depletion-load technology, and is packaged in a 40-pin plastic or ceramic DIP.

Fig. 1 - PIN FUNCTION

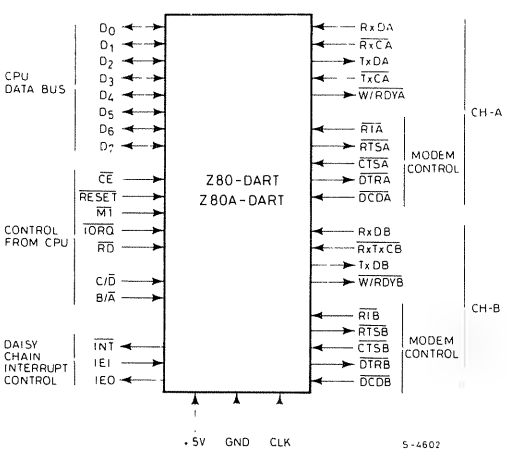
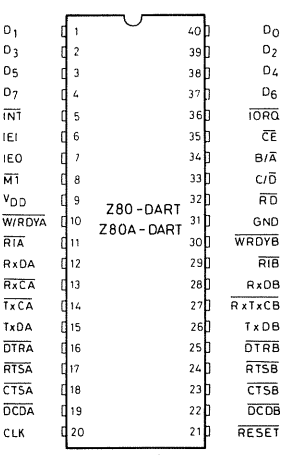


Fig. 2 - PIN ASSIGNMENTS



Pin Description

B/$\overline{A}$. *Channel A Or B Select* (input, High selects Channel B). This input defines which channel is accessed during a data transfer between the CPU and the Z80 DART.

C/$\overline{D}$. *Control Or Data Select* (input, High selects Control) This input specifies the type of information (control or data) transferred on the data bus between the CPU and the Z80 DART.

$\overline{CE}$. *Chip Enable* (input, active Low). A low at this input enables the Z80 DART to accept command or data input from the CPU during a write cycle, or to transmit data to the CPU during a read cycle.

CLK. *System Clock* (input). The Z80 DART uses the standard Z80 single-phase system clock to synchronize internal signals.

$\overline{CTSA}$, $\overline{CTSB}$. *Clear To Send* (inputs, active Low). When programmed as Auto Enables,a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accomodate slow-risetime signals.

D_0-D_7. *System Data Bus* (bidirectional, 3-state) transfers data and commands between the CPU and the Z80 DART.

$\overline{DCDA}$, $\overline{DCDB}$. *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if the Z80 DART is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered.

$\overline{DTRA}$, $\overline{DTRB}$. *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into the DTR bit. They can also be programmed as general-purpose outputs.

IEI. *Interrupt Enable In* (input, active High) is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this Z80 DART. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{INT}$. *Interrupt Request* (output, open drain, active Low). When the Z80 DART is requesting an interrupt, it pulls $\overline{INT}$ Low.

$\overline{M1}$. *Machine Cycle One* (input from Z80 CPU, active Low). When $\overline{M1}$ and $\overline{RD}$ are both active, the Z80 CPU is fetching an instruction from memory; when $\overline{M1}$ is active while $\overline{IORQ}$ is active, the Z80 DART accepts $\overline{M1}$ and $\overline{IORQ}$ as an interrupt aknowledge if the Z80 DART is the highest priority device that has interrupted the Z80 CPU.

$\overline{IORQ}$. *Input/Output Request* (input from CPU, active Low). $\overline{IORQ}$ is used in conjunction with B/$\overline{A}$, C/$\overline{D}$, $\overline{CE}$ and $\overline{RD}$ to transfer commands and data between the CPU and the Z80 DART. When $\overline{CE}$, $\overline{RD}$ and $\overline{IORQ}$ are all active, the channel selected by B/$\overline{A}$ transfers data to the CPU (a read operation). When $\overline{CE}$ and $\overline{IORQ}$ are active, but $\overline{RD}$ is inactive, the channel selected by B/$\overline{A}$ is written to by the CPU with either data or control information as specified by C/$\overline{D}$.

$\overline{RxCA}$, $\overline{RxCB}$. *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of $\overline{RxC}$. The Receive Clocks may be 1, 16, 32 or 64 times the data rate.

$\overline{RD}$. *Read Cycle Status.* (input from CPU, active Low). If $\overline{RD}$ is active, a memory or I/O read operation is in progress.

RxDA, RxDB. *Receive Data* (inputs, active High).

$\overline{RESET}$. *Reset* (input, active Low). Disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls High and disables all interrupts.

$\overline{RIA}$, $\overline{RIB}$. *Ring Indicator* (inputs, Active Low). These inputs are similar to $\overline{CTS}$ and $\overline{DCD}$. The Z80 DART detectes both logic level transitions and interrupts the CPU. When not used in switched-line applications, these inputs can be used as general-purpose inputs.

$\overline{RTSA}$, $\overline{RTSB}$. *Request to Send* (outputs, active Low). When the RTS bit is set, the $\overline{RTS}$ output goes Low. When the RTS bit is reset, the output goes High after the transmitter empties.

$\overline{TxCA}$, $\overline{TxCB}$. *Transmitter Clocks* (inputs). TxD changes on the falling of TxC. The Transmitter Clocks may be 1, 16, 32 or 64 times the data rate; however, the clock multiplier for the transmitter and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered. Both the Receiver and Transmitter Clocks may be driven by the Z80 CTC Counter Time Circuit for programmable baud rate generation.

TxDA, TxDB. *Transmit Data* (outputs, active High).

$\overline{W/RDYA}$, $\overline{W/RDYB}$. *Wait/Ready* (outputs, open drain when programmed for Wait function, driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the Z80 DART data rate. The reset state is open drain.

Functional Description

The functional capabilities of the Z80 DART can be described from two different points of view; as a data communications device, it transmits and receives serial data, and meets the requirements of asynchronous data communications protocols; as a Z80 family peripheral, it interacts with the Z80 CPU and other Z80 peripheral circuits, and shares the data, address and control buses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessor, the Z80 DART offers valuable features such as non-vectored interrupts, polling and simple handshake capability.

The first part of the following functional description introduces Z80 DART data communications capabilities; the second part describes the interaction between the CPU and the Z80 DART.

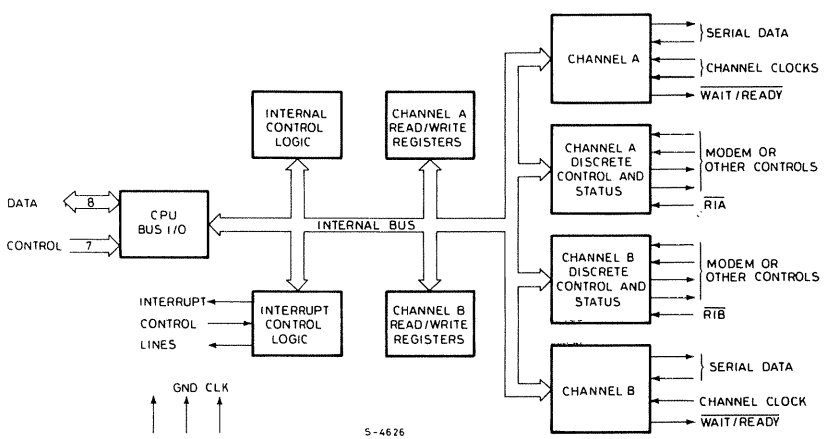
A more detailed explanation of Z80 DART operation can be found in the *Z80 SIO Technical Manual* (Document Number 03-3033-01). Because this manual was written for the Z80 SIO, it contains information about synchronous as well as asynchronous operation.

Communications Capabilities. The Z80 DART provides two independent full-duplex channels for use as an asynchronous receiver/transmitter. The following is a short description of receiver/transmitter capabilities. For more details, refer to the Asynchronous Mode section of the *Z80 SIO Technical Manual* The Z80 DART offers transmission and reception of five to eight bits per character, plus optional even or odd parity. The transmitter can supply one, one and a half or two stop bits per character and can provide a break output at any time. The receiver break detection logic interrupts the CPU both at the start and end of a received break. Reception is protected from spikes by a transient spike rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the Receive Data input. If the Low does not persist – as in the case of a transient – the character assembly process is not started.

Framing errors and overrung errors are detected and buffered together with the character on which they occurred. Vectored interrupts allow fast servicing of interrupting conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The Z80 DART does not require symmetric Transmit and Receive Clock signals – a feature that allows it to be used with a Z80 CTC or any other clock source. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32 or 1/64 of the clock rate supplied to the Receive and Transmit Clock inputs. When using Channel B, the bit rates for transmit and receive operations must be the same because $\overline{RxC}$ and $\overline{TxC}$ are bonded together ($\overline{RxTxCB}$).

Fig. 3 - BLOCK DIAGRAM



Functional Description (continued)

I/O Interface Capabilities. The Z80 DART offers the choice of Polling, Interrupt (vectored or non-vectored) and Block Transfer modes to transfer data, status and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

POLLING. There are no interrupts in the Polled mode. Status registers RR0 and RR1 are updated at appropriate times for each function being performed. All the interrupt modes of the Z80 DART must be disabled to operate the device in a Polled environment.

While in its Polling sequence, the CPU examines the status contained in RR0 for each channel, the RR0 status bits serve as an acknowledge to the Poll inquiry. The two RR0 status bits D_0 and D_2 indicate that a data transfer is needed. The status also indicates Error or other special status conditions (see "Z80 DART Programming"). The Special Receive Condition status contained in RR1 does not have to be read in a Polling sequence because the status bits in RR1 are accompanied by a Receive Character Available status in RR0.

INTERRUPTS. The Z80 DART offers an elaborate interrupt scheme that provides fast interrupt response in real-time applications. As a member of the Z80 family, the Z80 DART can be daisy-chained along with other Z80 peripherals for peripheral interrupt-priority resolution. In addition, the internal interrupts of the Z80 DART are nested to prioritize the various interrupts generated by Channels A and B. Channel B registers WR2 and RR2 contain the interrupt vector that points to an interrupt service routine in the memory. To eliminate the necessity of writing a status analysis routine, the Z80 DART can modify the interrupt vector in RR2 so it points directly to one of eight interrupt service routines. This is done under program control by setting a program bit (WR1, D_2) in Channel B called "Status Affects Vector" When this bit is set, the interrupt vector in RR2 is modified according to the assigned priority of the various interrupting conditions.

Transmit interrupts, Receive interrupts and External/Status interrupts are the main sources of interrupts. Each interrupts source is enabled under program control with Channel A having a higher priority than Channel B, and with Receiver, Transmit and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted by the transmit buffer *becoming* empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways:

- Interrupt on the first received character
- Interrupt on all received characters
- Interrupt on a Special Receive condition

Interrupt On First Character is typically used with the Block Transfer mode. Interrupt On All Receive Characters can optionally modify the interrupt vector in the event of a parity error. The Special Receive Condition interrupt can occur on a character basis. The Special Receive condition can cause an interrupt only if the Interrupt On First Receive Character or Interrupt On All Receive Characters mode is selected. In Interrupt On First Receive Character, an interrupt can occur from Special Receive conditions (except Parity Error) after the first receive character interrupt (example: Receive Overrun interrupt).

The main function of the External/Status interrupt is to monitor the signal transitions of the $\overline{CTS}$, $\overline{DCD}$ and $\overline{RI}$ pins; however, an External/Status interrupt is also caused by the detection of a Break sequence in the data stream. The interrupt caused by the Break sequence has a special feature that allows the Z80 DART to interrupt when the Break sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Break condition.

CPU/DMA BLOCK TRANSFER. The Z80 DART provides a Block Transfer mode to accomodate CPU block transfer functions and DMA block transfers (Z80 DMA or other designs). The Block Transfer mode uses the $\overline{W/RDY}$ output in conjunction with the Wait/Ready bits of Write Register 1. The $\overline{W/RDY}$ output can be defined under software control as a Wait line in the CPU Block Transfer mode or as a Ready line in the DMA Block Transfer mode.

To a DMA controller, the Z80 DART Ready output indicates that the Z80 DART is ready to transfer data to or from memory. To the CPU, the Wait output indicates that the Z80 DART is not ready to transfer data thereby requesting the CPU to extend the I/O cycle.

Internal Architecture

The device internal structure includes a Z80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains read and write registers, and discrete control and status logic that provides the interface to modems or other external devices.

The read and write register group includes five 8-bit control registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B that may be read through Read Register 2 in Channel B. The registers for both channels are designated as follows.

WR0-WR5 Write Registers 0 through 5
RR0-RR2 Read Registers 0 through 2

The bit assignment and functional grouping of each register is configured to simplify and organize the programming process.

Internal Architecture (continued)

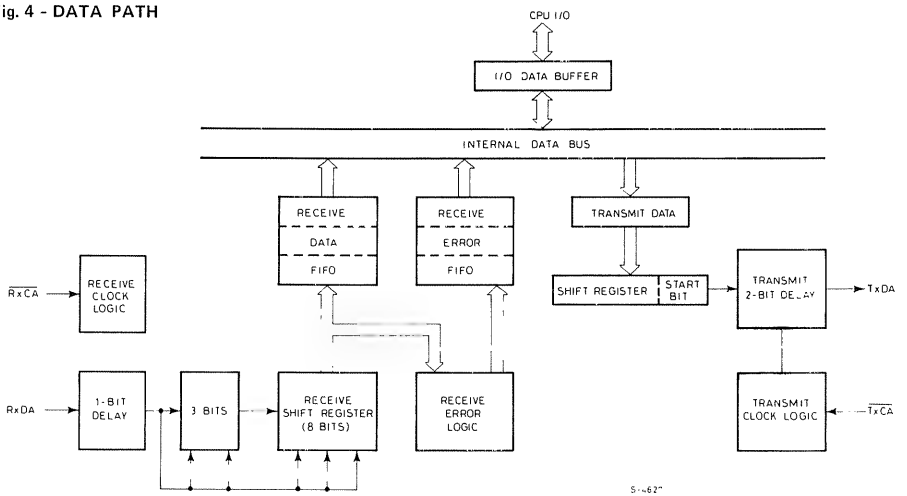
The logic for both channels provides formats, bit synchronization and validation for data transferred to and from the channel interface. The modem control inputs Clear to Send ($\overline{CTS}$). Data Carrier Detect ($\overline{DCD}$) and Ring Indicator ($\overline{RI}$) are monitored by the control logic under program control. All the modem control signals are general purpose in nature and can be used for functions other than modem control.

For automatic interrupt vectoring, the interrupt control logic determines which channel and which device within the channel has the highest priority. Priority is fixed with Channel A assigned a higher priority than Channel B; Receive, Transmit and External/Status interrupts are prioritized in that order within each channel.

Data Path. The transmit and receive data path illustrated for Channel A in Figure 4 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service a Receive Character Available interrupt in a high-speed data transfer.

The transmitter has an 8-bit transmit data register that is loaded from the internal data bus, and a 9-bit transmit shift register that is loaded from the transmit data register.

Fig. 4 - DATA PATH



Read, Write and Interrupt Timing

Read Cycle. The timing signals generated by a Z80 CPU input instructions to read a Data or Status byte from the Z80 DART are illustrated in Figure 5a.

Write Cycle. Figure 5b illustrates the timing and data signals generated by a Z80 CPU output instruction to write a Data or Control byte into the Z80 DART.

Interrupt Acknowledge Cycle. After receiving an Interrupt Request signal ($\overline{INT}$ pulled Low), the Z80 CPU sends an Interrupt Acknowledge signal ($\overline{M1}$ and $\overline{IORQ}$ both Low). The daisy-chained interrupt circuits determine the highest priority interrupt requestor. The IEI of the highest priority peripheral is terminated High. For any peripheral that has no interrupt pending or under service, IEO = IEI. Any peripheral that does have an interrupt pending or under service forces its IEO Low.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while $\overline{M1}$ is Low. When $\overline{IORQ}$ is Low, the highest priority interrupt requestor (the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

Refer to the *Z80 SIO Technical Manual* for additional details on the interrupt daisy chain and interrupt nesting.

Read, Write and Interrupt Timing (continued)

Return From Interrupt Cycle. Normally, the Z80 CPU issues an RETI (Return From Interrupt) instructions at the end of an interrupt service routine RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch to terminate the interrupt that has just been processed.

When used with other CPUs, the Z80 DART allows the user to return from the interrupt cycle with a special command called "Return From Interrupt" in Write Register 0 of Channel A. This command is interpreted by the Z80 DART in exactly the same way it would interpret an RETI command on the data bus

Fig. 5a - READ CYCLE

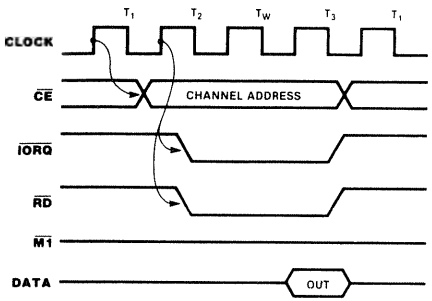


Fig. 5b - WRITE CYCLE

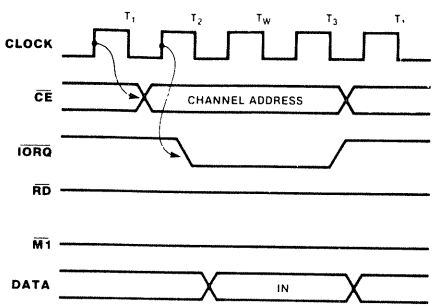


Fig. 5c - INTERRUPT ACKNOWLEDGE CYCLE

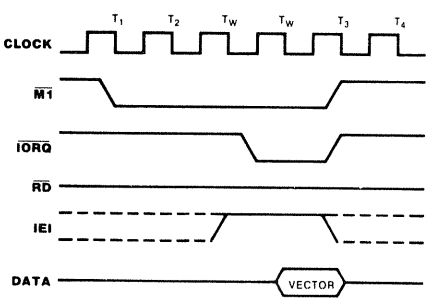
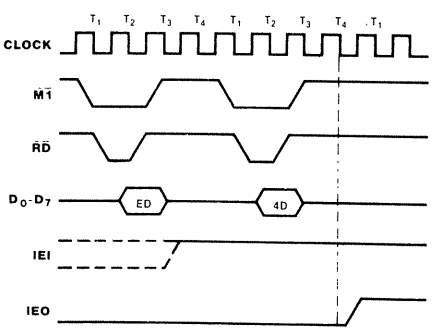


Fig. 5d - RETURN FROM INTERRUPT CYCLE



Z80 DART Programming

To program the Z80 DART, the system program first issues a series of commands that initialize the basic mode and then other commands that qualify conditions within the selected mode. For example, the character length, clock rate, number of stop bits, even or odd parity are first set, then the Interrupt mode and, finally, receiver or transmitter enable.

Both channels contain command registers that must be programmed via the system program prior to operation. The Channel Select input (B/$\overline{A}$) and the Control/Data input (C/$\overline{D}$) are the command structure addressing controls, and are normally controlled by the CPU address bus.

Writer Registers. The Z80 DART contains six registers (WR0–WR5) in each channel that are programmed separately by the system program to configure the functional personality of the channels (Figure 4). With the exception of WR0, programming the write registers requires two bytes. The first byte contains three bits (D_0-D_2) that point to the selected register; the second byte is the actual control word that is written into the register to configure the Z80 DART.

WR0 is a special case in that all the basic commands (CMD_0-CMD_2) can be accessed with a single byte. Reset (Internal or external) initializes the pointer bits D_0-D_2 to point to WR0. This means that a register cannot be pointed to in the same operation as a channel reset.

Write Register Functions	
WR0	Register pointers, initialization commands for the various modes, etc.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls

Read Registers. The Z80 DART contains three registers (RR0–RR2) that can be read to obtain the status information for each channel (except for RR2, which applies to Channel B only). The status information includes error conditions, interrupt vector and standard communications-interface-signals.

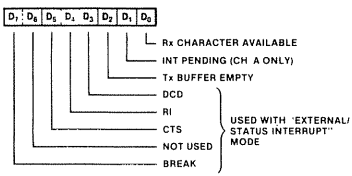
To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operations. Then, by executing an input instructions, the contents of the addressed read register can be read by the CPU.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Conditions interrupts has occurred, all the appropriate error bits can be read from a single register (RR1).

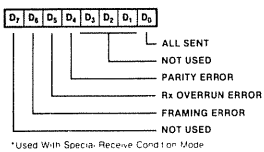
Read Register Functions	
RR0	Trasmit/Receiver buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)

Z80 DART Read and Write Registers

READ REGISTER 0

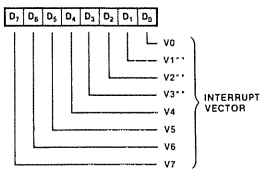


READ REGISTER 1*



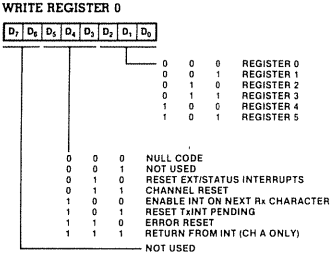
*Used With Special Receive Condition Mode

READ REGISTER 2

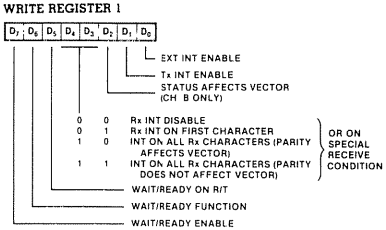


**Variable If Status Affects Vector Is Programmed

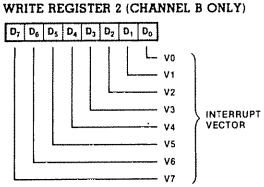
WRITE REGISTER 0



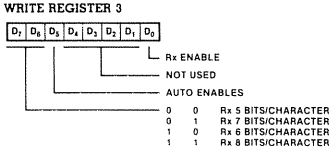
WRITE REGISTER 1



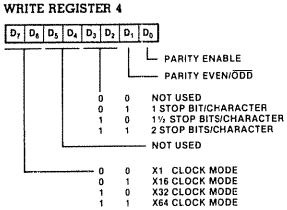
WRITE REGISTER 2 (CHANNEL B ONLY)



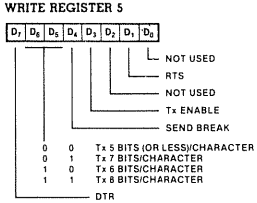
WRITE REGISTER 3



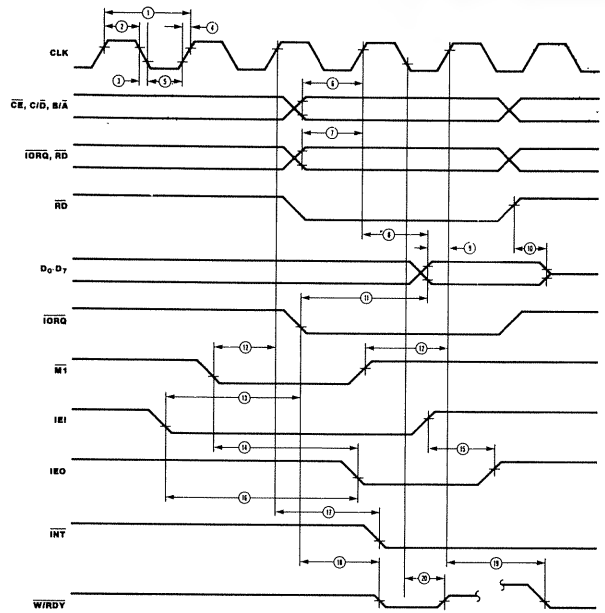
WRITE REGISTER 4



WRITE REGISTER 5

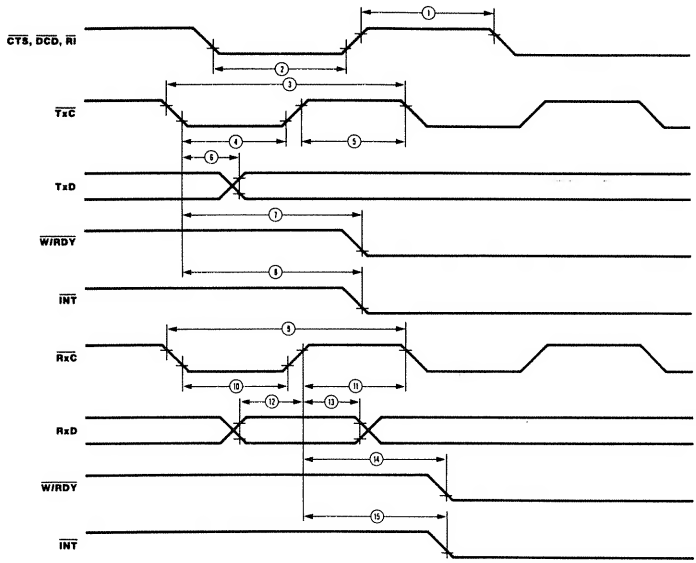


AC Electrical Characteristics



Number	Symbol	Parameter	Z80 DART		Z80A DART		Unit
			Min	Max	Min	Max	
1	TcC	Clock Cycle Time	400	4000	250	4000	ns
2	TwCh	Clock Width (High)	170	2000	105	2000	ns
3	TfC	Clock Fall Time		30		30	ns
4	TrC	Clock Rise Time		30		30	ns
5	TwCl	Clock Width (Low)	170	2000	105	2000	ns
6	TsAD(C)	$\overline{CE}$, C/$\overline{D}$, B/$\overline{A}$ to Clock ↑ Setup Time	160		145		ns
7	TsCS(C)	$\overline{IORQ}$, $\overline{RD}$ to Clock ↑ Setup Time	240		115		ns
8	TdC(DO)	Clock ↑ to Data Out Delay		240		220	ns
9	TsDI(C)	Data In to Clock ↑ Setup Time	50		50		ns
10	TdRD(DOz)	$\overline{RD}$ ↑ to Data Out Float Delay		230		110	ns
11	TdIO(DOI)	$\overline{IORQ}$ ↑ to Data Out Delay (INTA Cycle)		340		160	ns
12	TsM1(C)	$\overline{M1}$ To Clock ↑ Setup Time	210		90		ns
13	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTA Cycle)	200		140		ns
14	TdM1 (IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt immediately preceding $\overline{M1}$ ↓)		300		190	ns
15	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED decode)		150		100	ns
16	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		150		100	ns
17	TdC(INT)	Clock ↑ to INT ↓ Delay		200		200	ns
18	TdIO(W/RWf)	$\overline{IORQ}$ ↓ or $\overline{CE}$ ↓ to $\overline{W/RDY}$ ↓ Delay (Wait Mode)		300		210	ns
19	TdC(W/RR)	Clock ↑ to $\overline{W/RDY}$ ↓ Delay (Ready Mode)		120		120	ns
20	TdC(W/RWz)	Clock ↓ to $\overline{W/RDY}$ Float Delay (Wait Mode)		150		130	ns

AC Electrical Characteristics (continued)



Number	Symbol	Parameter	Z80 DART		Z80A DART		Unit
			Min	Max	Min	Max	
1	TwPh	Pulse Width (High)	200		200		ns
2	TwPl	Pulse Width (Low)	200		200		ns
3	TcTxC	$\overline{TxC}$ Cycle Time	400	∞	400	∞	ns
4	TwTxCl	$\overline{TxC}$ Width (Low)	180	∞	180	∞	ns
5	TwTxCh	$\overline{TxC}$ Width (High)	180	∞	180	∞	ns
6	TdTxC(TxD)	$\overline{TxC}$ ↓ to TxD Delay		400		300	ns
7	TdTxC(W/RRf)	$\overline{TxC}$ ↓ to $\overline{W/RDY}$ ↓ Delay (Ready Mode)	5	9	5	9	Clk Per.
8	TdTxC(INT)	$\overline{TxC}$ ↓ to $\overline{INT}$ ↓ Delay	5	9	5	9	Clk Per.
9	TcRxC	$\overline{RxC}$ Cycle Time	400	∞	400	∞	ns
10	TwRxCl	$\overline{RxC}$ Width (Low)	180	∞	180	∞	ns
11	TwRxCh	$\overline{RxC}$ Width (High)	180	∞	180	∞	ns
12	TsRxD(RxC)	RxD to $\overline{RxC}$ ↑ Setup Time (xl Mode)	0		0		ns
13	ThRxD(RxC)	RxD Hold time (xl Mode)	140		140		ns
14	TdRxC(W/RRf)	$\overline{RxC}$ ↑ to $\overline{W/RDY}$ ↓ Delay (Ready Mode)	10	13	10	13	Clk Per.
15	TdRxC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ ↓ Delay	10	13	10	13	Clk Per.

In all modes, the Clock rate must be at least five times the maximum data rate.
$\overline{RESET}$ must be active a minimum of one complete Clock Cycle.

Absolute Maximum Ratings*

Voltages on all inputs and outputs with respect to GND	-0.3 to +7.0 V
Operating Ambient Temperature	As Specified in Ordering Information
Storage Temperature	-65 to +150 °C

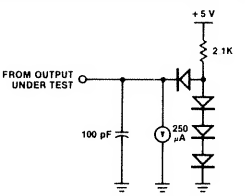
* Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Standard Test Conditions

The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

- +4.75V ≤ V_{CC} ≤ +5.25 V
- GND = 0 V
- T_A as specified in Ordering Information

All ac parameters assume a load capacitance of 100 pF max. Timing references between two output signals assume a load difference of 50 pF max.

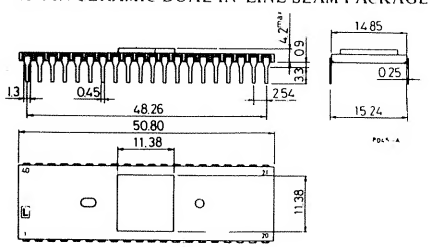


DC Characteristics $T_A = 0°C$ to 70°C, $V_{CC} = +5V, \pm 5\%$

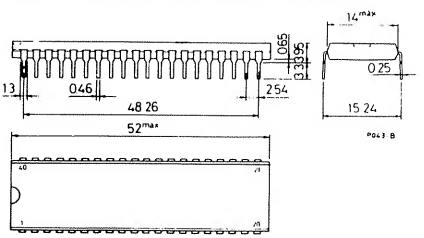
Symbol	Parameter	Min	Max	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3	+0.45	V	
V_{IHC}	Clock Input High Voltage	V_{CC} -0.6	+5.5	V	
V_{IL}	Input Low Voltage	-0.3	+0.8	V	
V_{IH}	Input High Voltage	+2.0	+5.5	V	
V_{OL}	Output Low Voltage		+0.4	V	I_{OL} = 2.0 mA
V_{OH}	Output High Voltage	+2.4		V	I_{OH} = -250 µA
I_L	Input/3-State Output Leakage Current	-10	+10	µA	0.4 < V < 2.4V
$I_{L(RI)}$	RI Pin Leakage Current	-40	+10	µA	0.4 < V < 2.4V
I_{CC}	Power Supply Current		100	mA	

MECHANICAL DATA (dimensions in mm)

40-PIN CERAMIC DUAL IN-LINE SLAM PACKAGE



40-PIN PLASTIC DUAL IN-LINE PACKAGE



ORDERING NUMBERS:

Z80 DART D1 for dual in-line ceramic package
Z80 DART B1 for dual in-line plastic package
Z80A DART D1 for dual in-line ceramic package
Z80A DART B1 for dual in-line plastic package

SGS-ATES GROUP OF COMPANIES

INTERNATIONAL HEADQUARTERS
SGS-ATES Componenti Elettronici SpA
Via C. Olivetti 2 - 20041 Agrate Brianza - Italy
Tel.: 039 - 65551
Telex: 330131-330141

BENELUX
SGS-ATES Componenti Elettronici SpA
Benelux Sales Office
B- 1180 Bruxelles
Winston Churchill Avenue, 122
Tel.: 02 - 3432439
Telex: 24149 B

DENMARK
SGS-ATES Scandinavia AB
Sales Office:
2730 Herlev
Herlev Torv 4
Tel.: 02 - 948533
Telex: 35411

EASTERN EUROPE
SGS-ATES Componenti Elettronici SpA
Export Sales Office
20041 Agrate Brianza - Italy
Via C. Olivetti, 2
Tel.: 039 - 6555287/6555207
Telex: 330131-330141

FINLAND
Sales Office:
SGS-ATES Scandinavia AB
02210 Esbo 21
Kääntöpiiri 2
Tel.: 90 - 881395/6
Telex: 123643

FRANCE
SGS-ATES France S.A.
75643 Paris Cedex 13
Résidence "Le Palatino"
17, Avenue de Choisy
Tel.: 5842730
Telex: 042 - 250938

GERMANY
SGS-ATES Deutschland Halbleiter
Bauelemente GmbH
8018 Grafing bei München
Haidling 17
Tel.: 08092-691
Telex: 05 27378
Sales Offices:
3012 Langenhagen
Hubertusstrasse 7
Tel.: 0511 - 772075/7
Telex: 09 23195
8000 München 21
Landsbergerstrasse 289
Tel.: 089 - 582047/8
Telex: 05 215784
8500 Nürnberg 15
Parsifalstrasse 10
Tel.: 0911 - 49645/6
Telex: 0626243
7000 Stuttgart 80
Kalifenweg 45
Tel.: 0711 - 713091/2
Telex: 07 255545

HONG KONG
SGS-ATES Singapore (Pte) Ltd.
1329 Ocean Centre
Canton Road, Kowloon
Tel.: 3 - 662625
Telex: ESGIE HK 63906

ITALY
SGS-ATES Componenti Elettronici SpA
Direzione Commerciale Italia
20149 Milano
Via Correggio, 1/3
Tel.: 02 - 4695651
Sales Offices:
50127 Firenze
Via Giovanni Del Pian Dei Carpini 96/1
Tel.: 055 - 4377763
20149 Milano
Via Correggio 1/3
Tel.: 02 - 4695651
00199 Roma
Piazza Gondar, 11
Tel.: 06 - 8392848/8312777
10121 Torino
Corso G. Ferraris, 26
Tel.: 011-531167

SINGAPORE
SGS-ATES Singapore (Pte) Ltd.
Singapore 1231
Lorong 4 & 6 - Toa Payoh
Tel.: 2531411
Telex: ESGIES RS 21412

SWEDEN
SGS-ATES Scandinavia AB
19501 Märsta
Box 144
Tel.: 0760 - 40120
Telex: 042 - 10932

UNITED KINGDOM
SGS-ATES (United Kindgom) Ltd.
Aylesbury, Bucks
Planar House, Walton Street
Tel.: 0296 - 5977
Telex: 041-83245

U.S.A.
SGS-ATES Semiconductor Corporation
Scottsdale, AZ85251
7070 East 3rd Avenue
Tel.: (602) 990-9553
Telex: SGS ATES SCOT 165808
Waltham, MA 02154
240 Bear Hill Road
Tel.: (617) 890-6688
Telex: 923495 WHA
Des Plaines, IL 60018
2340 Des Plaines Ave Suite 309
Tel.: (312) 296-4035
Telex: 282547

SGS-ATES GROUP OF COMPANIES
Italy – France – Germany – Singapore – Sweden – United Kingdom – U.S.A.

1981 - Printed in Italy by permission of Zilog Inc.

Suggested price
U.S. $ 3 (including tax)

ORDER CODE: DAZ80MSYS/3